IMPLEMENTASI SISTEM OTOMATISASI PINTU DENGAN *FACE RECOGNITION* MENGGUNAKAN METODE *HAAR-CASCADE* DAN *LOCAL BINARY PATTERN* PADA RASPBERRY PI

SKRIPSI

KEMINATAN TEKNIK KOMPUTER

Untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Komputer

Disusun oleh:
Willy Andika Putra
135150300111042

PROGRAM STUDI TEKNIK INFORMATIKA
JURUSAN TEKNIK INFORMATIKA
FAKULTAS ILMU KOMPUTER
UNIVERSITAS BRAWIJAYA
MALANG
2018

PENGESAHAN

IMPLEMENTASI SISTEM OTOMATISASI PINTU DENGAN *FACE RECOGNITION* MENGGUNAKAN METODE *HAAR-CASCADE* DAN *LOCAL BINARY PATTERN* PADA RASPBERRY PI

SKRIPSI

KEMINATAN TEKNIK KOMPUTER

Diajukan untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Komputer

Disusun Oleh :
Willy Andika Putra
135150300111042

Skripsi ini telah diuji dan dinyatakan lulus pada
30 Juli 2018
Telah diperiksa dan disetujui oleh:

Dosen Pembimbing I	Dosen Pembimbing II
Rizal Maulana, S.T., M.T., M.Sc.	Fitri Utaminingrum, Dr. Eng., S.T,M.T
NIK. 201607 891009 1 001	NIP. 19820710 200812 2 001

Mengetahui
Ketua Jurusan Teknik Informatika

Tri Astoto Kurniawan, S.T, M.T, Ph.D
NIP. 197105182003121001

PERNYATAAN ORISINALITAS

Saya menyatakan dengan sebenar-benarnya bahwa sepanjang pengetahuan saya, di dalam naskah skripsi ini tidak terdapat karya ilmiah yang pernah diajukan oleh orang lain untuk memperoleh gelar akademik di suatu perguruan tinggi, dan tidak terdapat karya atau pendapat yang pernah ditulis atau diterbitkan oleh orang lain, kecuali yang secara tertulis disitasi dalam naskah ini dan disebutkan dalam daftar pustaka.

Apabila ternyata didalam naskah skripsi ini dapat dibuktikan terdapat unsur-unsur plagiasi, saya bersedia skripsi ini digugurkan dan gelar akademik yang telah saya peroleh (sarjana) dibatalkan, serta diproses sesuai dengan peraturan perundang-undangan yang berlaku (UU No. 20 Tahun 2003, Pasal 25 ayat 2 dan Pasal 70).

Malang, 13 Juni 2018

Willy Andika Putra

NIM: 135150300111042

KATA PENGANTAR

Puji Syukur penulis panjatkan kepada Tuhan Yang Maha Esa karena hanya dengan kasih sayang dan anugerahNya penulis dapat menyelesaikan skripsi dengan judul Implementasi Sistem Otomatisasi Pintu Dengan *Face Recognition* Menggunakan Metode *Haar-Cascade* Dan *Local Binary Pattern* Pada Raspberry Pi.

Melalui kesempatan ini, penulis ingin menyampaikan terima kasih yang sebesar-besarnya kepada semua pihak yang telah memberikan bantuan dan dukungan selama proses penulisan skripsi ini hingga selesai, diantaranya:

1. Bapak Rizal Maulana,S.T.,M.T.,M.Sc. dan Ibu Fitri Utaminingrum, Dr. Eng.,S.T,M.T selaku dosen pembimbing skripsi yang telah memberikan masukan dan ilmu serta saran dalam menyelesaikan skripsi ini.
2. Bapak Tri Astoto Kurniawan, S.T, M.T, Ph. D selaku Ketua Jurusan Teknik Informatika Fakultas Ilmu Komputer Universitas Brawijaya Malang.
3. Bapak Sabriansyah Rizqika Akbar,S.T.,M.Eng. selaku Ketua Program Studi Teknik Komputer Fakultas Ilmu Komputer Universitas Brawijaya Malang.
4. Kedua orang tua Bapak Hamdi Sehan dan Ibu Dra. Rumsiah dan seluruh keluarga besar yang telah memberikan dukungan dan bantuan baik bantuan berupa doa maupun bantuan moral.
5. Saudara kandung Friska Andiani, Fani Andiesta, Aditya Ilhamdi Reza yang selalu mendoakan dan memberi semangat setiap saat.
6. Silvia Devi Enggarwati yang selalu mau mendengarkan keluh kesah mengenai skripsi dan memberikan bantuan berupa saran dan masukan kepada penulis.
7. Teman-teman Kampung Lebak Haur yang telah membantu memberikan saran kepada penulis sepanjang pengerjaan hingga skripsi ini selesai.
8. Teman-teman terdekat Gagah Ragil Triatmojo, M. Hajriantoso, Tri Umpu Kiraton, Dina Aulia yang sudah lulus terlebih dahulu dan bisa memberikan tekanan kepada penulis untuk segera menyelesaikan skripsi ini.
9. Temen-teman “BKK 45” Kresna, Fajar, Indera, Khulil, Rudy, Delta, Noor, Rizki, Alfian, Aji, Hendra, Tegar, Adnan, Bagus, Zamroni, Bily, Yogi, Galih yang memberikan motivasi sepanjang pengerjaan.
10. Teman-teman Sistem Komputer angkatan 2013 yang selalu mendukung dan berbagi ilmu dari awal perkuliahan sampai tahap akhir penyelesaian skripsi.
11. Seluruh civitas akademika Fakultas Ilmu Komputer Universitas Brawijaya yang telah banyak memberi bantuan dan dukungan selama penyelesaian skripsi ini.

12. Semua pihak yang telah membantu dan berbagi ilmu dalam penyelesaian skripsi ini yang tidak dapat penulis sebutkan satu per satu.

Penulis menyadari bahwa dalam penyusunan skripsi ini masih banyak kekurangan, sehingga saran dan kritik yang membangun sangat penulis harapkan. Akhir kata penulis berharap skripsi ini dapat membawa manfaat bagi semua pihak yang menggunakannya.

Malang, 13 Juni 2018

Penulis

willy2214@gmail.com

ABSTRAK

Perkembangan teknologi di dunia saat ini mulai merambah menuju era *IOT*(*Inthernet Of Things*). Teknologi IOT yang mulai dirasakan dampaknya oleh banyak orang salah satunya ialah *Smarthome. Smarthome* adalah sebuah hunian yang ditanamkan banyak teknologi yang terintegrasi dan dapat membuat hampir semua aspek dalam rumah berjalan otomatis. Salah satu aspek yang perlu diperhatikan pada *smarthome* ialah aspek keamanan, sudah seharusnya semua hunian yang terintegrasi *smarthome* memiliki sistem keamanan yang sangat cukup. Sistem keamanan yang dimiliki oleh *smarthome* sebagian besar diimplementasikan pada pintu, misalnya menanamkan sebuah sensor pada pintu agar pintu dapat terkunci secara otomatis. Namun, akan ada banyak kendala ketika menggunakan teknologi sensor salah satunya ialah kebanyakan sensor tidak dapat mengidentifikasikan seseorang yang berada didekat pintu. Dari masalah tersebut akan dibuat suatu sistem otomatisasi buka tutup pintu pada *smarthome* yang dapat mengidentifikasikan seseorang. Teknologi identifikasi yang akan digunakan pada sistem ialah *Face Recognition* menggunakan metode *Haar-Cascade* yang akan di implementasikan pada *single-board circuit* yaitu *Raspberry Pi*. Sistem akan menyimpan identitas penghuni *smarthome* dalam sebuah *dataset* dan hanya akan membukakan pintu jika data sesuai dengan dataset yang telah dibuat sebelumnya. *Face Recognition* disini akan menggunakan *Library Open CV* yaitu *Haarcascade_Frontal_Face* dan *cv2.createLBPHFaceRecognizer*. Tingkat akurasi deteksi dari penggunaan metode *Haarcascade Classifier* ialah sebsar 76.25% sedangkan untuk tingkat akurasi pengenalan menggunakan *Local Binary Pattern* ialah sebesar 65%. Untuk Menghubungkan *face recognition* dengan sistem buka tutup pintu penulis menggunakan 3 buah motor DC. Dua buah motor DC akan betindak sebagai Roda Pintu dan 1 buah motor DC akan bertindak sebagai kunci. Semua proses akan dilakukan *Self Process* pada *Raspberry Pi*.

Kata Kunci : Otomatisasi Pintu, Raspberry Pi, *Face Detection, Face Recognition, Haar-Cascade Classifier.*

ABSTRACT

Technological developments in the world are starting towards the IOT (Inthernet Of Things) era. Smarthome is one of IOT Technology whose the impact is felt by many people. Smarthome is an embedded dwelling of many integrated technologies and make almost aspects in the home run automatically. One aspect that needs to be considered in the smarthome is the security aspect, all the smarthome integrated residence should has a very adequate security system. The security system owned by smarthome is mostly implemented on the door, for example implanting a sensor on the door to keep the door locked automatically. However, there will be many obstacles when using sensor technology, one of them is most sensors can not identify someone who is near the door. From the problem will be made an open door automation system on smarthome that can identify someone. The identification technology that will be used in the system is Face Recognition using Haar-Cascade method which will be implemented on single-board circuit that is Raspberry Pi. The system will store the identity of the Smarthome inhabitants in a dataset and will open the door if the data matches with the previously created dataset. Face Recognition will use Library Open CV that is Haarcascade_Frontal_Face and cv2.createLBPHFaceRecognizer . The level of detection accuracy using haarcascade classifier method is 76.25% while for the level of accuracy of introduction used local binary pattern is equal to 65%. To Connect face recognition with open door closing system, the author uses 3 pieces of DC motor. Two DC motors as Door Wheels and 1 DC motor as a key. All process will be done by Self Process on Raspberry Pi.

Keywords : Automatic Doors, Raspberry Pi, Face Detection Face Recognition, Haar-Cascade Classifier.

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

BAB I PENDAHULUAN

1.1 Latar Belakang

Perkembangan teknologi dan informasi dewasa ini semakin cepat dan semakin pesat. Berbagai teknologi baru mulai bermunculan mulai dari teknologi untuk tempat tinggal/*Smarthome* hingga perangkat elektronik yang bisa memberikan keamanan kepada yang menggunakannya. Salah satu contoh perangkat yang dapat memberikan rasa aman kepada penggunanya ialah teknologi CCTV. *Closed circuit television* (CCTV) merupakan alat perekaman yang menggunakan satu atau lebih kamera *video* dan menghasilkan data *video* atau *audio*. CCTV memiliki fungsi untuk merekam segala aktifitas dari jarak jauh tanpa batasan jarak, dapat memantau dan merekam segala bentuk aktifitas yang terjadi dilokasi pengamatan dengan menggunakan laptop atau PC secara *real time* dari mana saja, dan dapat merekam seluruh kejadian secara 24 jam, atau dapat merekam ketika terjadi gerakan dari daerah yang terpantau (Surjono, 1996)

Teknologi CCTV memanfaatkan kamera untuk merekam segala aktifitas namun teknologi CCTV hanya sebatas monitoring saja. CCTV tidak dapat menindak lebih lanjut ketika terjadi perampokan atau pencurian. Ada beberapa sistem keamanan yang memilik fungsi lebih yaitu untuk mencegah terjadinya tindakan pencurian salah satunya ialah sistem yang dapat mengontrol buka tutup kunci pintu melalui jaringan seluler. Sistem ini dapat mengontrol kunci pintu dengan cara memasukan kata kunci melalui *keypad* yang disediakan atau bisa juga melalui *handphone* yang kemudian diterima oleh modem GSM untuk selanjutnya memberikan perintah memasukan kata kunci pada mikrokontroller. Jika kata kunci salah sebanyak 3 kali maka alarm akan berbunyi dan jika kata kunci benar maka kunci akan terbuka (Rahajoeningroem & Wahyudin, 2013).

Kunci memang memegang peran penting dalam sebuah sistem keamanan. Pada sistem keamanan yang ditulis oleh (Rahajoeningroem & Wahyudin, 2013), sistem keamanan yang dibuat sudah cukup efektif untuk mengamankan sebuah rumah karena sudah dapat dikontrol melalui jarak jauh. Karena sistem menggunakan jaringan seluler dan tidak menggunakan identifikasi pengguna, maka akan muncul beberapa permasalahan. Permasalahan yang pertama ialah : Bagaimana jika ada seseorang yang kita kenal memaksa masuk ke dalam rumah namun belum mengetahui sistem keamanan yang digunakan ?. Dan permasalahan yang kedua ialah : Karena sistem menggunakan jaringan seluler untuk fungsi notifikasi, bagaimana jika pulsa pada modem GSM habis ?.

Dari permasalahan yang telah dipaparkan diatas, diperlukan sebuah sistem keamanan yang lebih handal untuk mencegah hal-hal tersebut. Sistem keamanan yang dapat mengidentifikasi pengguna dan juga sistem keamanan yang dapat bekerja tanpa jaringan seluler. Ada banyak cara yang dapat dilakukan untuk

mengidentifikasikan seseorang. Salah satu sistem identifikasi yang cukup popular ialah identifikasi menggunakan wajah. Kita dapat mengenali ribuan wajah karena frekuensi interaksi yang sangat sering ataupun hanya sekilas bahkan dalam rentang waktu yang sangat lama. Bahkan kita mampu mengenali seseorang walaupun terjadi perubahan pada orang tersebut karena bertambahnya usia atau pemakaian kacamata atau perubahan gaya rambut. Oleh karena itu wajah digunakan sebagai organ dari tubuh manusia yang dijadikan indikasi pengenalan seseorang atau *face recognition* (Sepritahara, 2012). *Face recognition* merupakan cara identifikasi yang tepat untuk diimplementasikan pada sistem. Ada banyak metode yang dapat digunakan untuk identifikasi wajah salah satu metode yang sering digunakan ialah *Haar-Cascade Classifier. Haar-Cascade Classifier* merupakan metode yang mudah untuk diimplementasikan dan dapat meng-*handle* data dalam jumlah besar, *Haar-Cascade Classifier* merupakan pendeteksi terbaik dalam kecepatan dan kehandalan (Singh, et al., 2013).

Dengan mengkombinasikan fungsi pengenalan wajah dengan sistem buka tutup pintu secara otomatis maka akan dapat menjawab permasalahan yang muncul pada penelitian sebelumnya. Sistem ini nantinya akan ditanamkan dalam sebuah mikrokontroller yaitu Raspberry Pi dan sebuah kamera berukuran kecil untuk meringkas ukuran perangkat.

1.2 Rumusan Masalah

Berdasarkan latar belakang diatas, ada beberapa permasalahan yang perlu dipecahkan, antara lain ialah :

1. Bagaimana cara mengimplementasikan *face recognition* ke dalam raspberry pi dengan menggunakan metode *haar-cascade classifier* ?
2. Seberapa besar tingkat akurasi deteksi wajah menggunakan metode *haar-cascade classifier* ?
3. Seberapa besar tingkat akurasi *face recognition* menggunakan *haar-cascade classifier* ?
4. Bagaimana cara menghubungkan *face recognition* dengan sistem buka tutup pintu rumah ?

1.3 Tujuan Penelitian

Berdasarkan latar belakang dan rumusan masalah, Ada beberapa tujuan yang harus dicapai dalam penelitian ini, antara lain ialah :

1. Dapat mengimplementasikan *face recognition* dengan metode haar-cascade ke dalam raspberry pi.
2. Dapat mengetahui seberapa besar akurasi deteksi wajah menggunakan metode *haar-cascade classifier*.
3. Dapat mengetahui seberapa besar akurasi *face recognition* menggunakan metode *haar-cascade classifier*.

4. Dapat menghubungkan *face recognition* dengan sistem buka tutup pintu rumah.

1.4 Manfaat Penelitian

Adapun manfaat yang dapat diberikan setelah penelitian ini dilakukan antara lain.

1. Mengaplikasikan ilmu yang diperoleh selama mengikuti perkuliahan di Sistem Komputer Universitas Brawijaya.
2. Membantu pengembangan sistem keamanan pada *Smarthome.*
3. Menambah fungsionalitas raspberry pi sebagai mikrokontroller.
4. Menambah keamanan dengan pengaplikasian *face recognition* pada sistem buka tutup pintu.

1.5 Batasan Masalah

Agar mendapatkan hasil yang sesuai maka dibutuhkan suatu batasan masalah, beberapa batasan yang akan diterapkan antara lain :

1. *Face recognition* dilakukan satu per satu didepan kamera sistem.
2. Prototipe sistem dibuat menggunakan kertas PVC dengan ukuran pintu 30 cm x 45 cm dan kunci berbentuk kail yang dapat berputar 180°.
3. *Face recognition* dilakukan pada 20 individu yang berbeda dengan maksimal 10 sampel foto maksimal per individu yang diambil dari depan.
4. Pengambilan sampel yang akan dijadikan *dataset* dilakukan dengan pencahayaan yang cukup.
5. *Audio* yang digunakan dalam pengenalan wajah dibuat menggunakan *FL Studio* 12 dan dianggap sudah siap pakai, jadi dalam penelitian tidak dibahas mengenai cara pembuatan *audio*.
6. Sistem tidak digunakan untuk mendeteksi dan mengenali foto.

1.6 Sistematika Penulisan

Dalam penulisan proposal ini dibagi menjadi beberapa bagian. Hal ini dilakukan agar pembaca dapat memahami dengan mudah isi dan maksud dari proposal penelitian ini. Bagian-bagian yang dimaksud ialah :

BAB 1 Pendahuluan

Dalam pendahuluan hal yang dibahas ialah permasalahan yang terjadi dan masalah apa yang akan dipecahkan. Dalam bab pendahuluan dibagi menjadi beberapa sub-bab yan saling berhubungan satu sama lain. Sub bab yang ada pada bab pendahuluan antara lain : latar belakang, rumusan masalah, tujuan penelitian, manfaat penelitian, batasan masalah dan sistematika penulisan.

BAB 2 Landasan Kepustakaan

Pada bab 2 akan berisi kajian pustaka dan dasar teori, Bab ini menjelaskan tentang pustaka-pustaka yang mendukung penelitian, memperkuat latar belakang

dan juga menjelaskan bahwa masalah yang diangkat sangatlah penting. Pada bab ini pula terdapat dasar teori yang dapat mendukung dalam perancangan sistem yang dimaksud pada bab 1. Dasar teori diambil dari penelitian yang sudah ada, penulis wajib mencantumkan setiap sumber yang diambil dari penelitian terdahulu.

BAB 3 Metodologi

Bab ini menjelaskan metodologi apa yang digunakan pada penelitian ini. Metode disini sangat penting karena hal ini bisa jadi membedakan antara penelitian dalam proposal ini dengan penelitian yang ada pada proposal lain. Metode juga mencangkup semua hal yang berkaitan dengan penelitian ini mulai dari pengumpulan data hingga pembuatan sistem.

BAB 4 Rekayasa Dan Kebutuhan Sistem

Bab ini menjelaskan apa sajakah yang dibutuhkan oleh sistem mulai dari kebutuhan secara perangkat lunak maupun secara perangkat keras. Dan juga dijelaskan bagaimana rekayasa kebutuhan secara fungsional maupun non fungsional.

BAB 5 Perancangan Dan Implementasi

Bab ini menjelaskan bagaimana cara pengimplementasian sistem secara terstruktur mulai dari tahap perancangan hingga sistem siap untuk dilakukan pengujian, pada bab ini pula akan dijelasakan bagaimana sistem dapat berjalan sebagaimana yang diinginkan pada bab 1.

BAB 6 Hasil Dan Analisis

Pada hasil penelitian akan menjelaskan secara rinci hasil yang didapat ketika melakukan penelitian dan hasilnya bisa berupa sesuai dengan bab 1 bisa juga tidak. Hal ini perlu ditulis dengan sangat jelas, agar tidak terjadi ambiguitas dalam penarikan kesimpulan.

BAB 7 Kesimpulan.

Dan yang terakhir ialah kesimpulan. Bab yang akan menjelaskan benang merah dari penelitian dalam proposal ini. Kesimpulan akan ditarik berdasarkan inti dari setiap bab agar kesimpulan sesuai dengan apa yang dibicarakan dalam proposal penelitian ini.

BAB II LANDASAN KEPUSTAKAAN

2.1. Tinjauan Pustaka

Dalam pembuatan tugas akhir ini diperlukan pustaka pembanding atau referensi yang berhubungan dengan pembuatan sistem. Setiap pustaka harus sesuai dan berkesinambungan antara pustaka yang satu dengan yang lainnya juga dengan tugas akhir ini. Terdapat 3 jurnal penelitian yang dapat menjadi pustaka pembanding dalam penelitian ini.

Penelitian pertama ialah tentang Sistem Buka Tutup Pintu Otomatis Menggunakan Jam Tangan Berbasis Mikrokontroller. Pada penelitian ini membahas tentang sistem keamanan yang dapat dikendalikan menggunakan jam tangan yang terbuat dari mikrokontroller arduino yang dikombinasikan menggunakan sensor IR. Penggerak pintu otomatis hanya dapat bekerja jika sensor IR yang ditempatkan pada bagian atas pintu atau bagian samping pintu menangkap sinar IR yang ada pada jam tangan. Dan sistem akan tetap bekerja jika jam tangan dipakai oleh seseorang yang bukan penghuni rumah (Permana & Dwiyono, 2015).

Penelitian kedua merupakan penelitian tentang Sistem Pengenalan Wajah Dengan Metode *Eigenface* Dan Jaringan Syaraf Tiruan (JST). Pada penelitian ini dibahas bagaimana cara sistem mengenali wajah dengan memadukan dua metode yaitu *Eigenface* dan juga Jaringan Syaraf Tiruan. Sistem memadukan kedua metode dengan tujuan agar akurasi dalam pengenalan wajah menjadi semakin tinggi. Sistem yang dibuat diimplementasikan pada personal komputer dengan sistem operasi windows, karena membutuhkan komputasi yang cukup besar agar dapat berjalan (Mulyono, et al., 2012). Sistem pengenalan wajah diatas tidak mungkin untuk diimplementasikan pada sebuah pintu karena menggunakan komputer desktop yang memiliki ukuran besar.

Penelitian ketiga ialah penelitian tentang Penerapan *Face Recognition* dengan Metode *Eigenface* pada *Intelligent Car Security.* Pada penelitian yang dilakukan oleh (Sehman, 2015) sistem pengenalan wajah sudah diterapkan pada sebuah mikrokontroller. *Face recognition* diimplementasikan pada sistem penguncian mobil guna menghindari tindak pencurian. Sistem akan men-*trigger* sebuah alarm jika wajah yang terdeteksi tidak dikenali. Sistem ini berfokus pada bagaimana menghubungkan antara *Face Recognition* dengan sistem penguncian yang ada pada sebuah mobil (Sehman, 2015). Pada penelitian ini penggunaan sensor untuk membunyikan alarm pada mobil dirasa kurang efektif. Oleh karena itu perlu pembaruan berupa penanaman otomatisasi pintu mulai dari buka tutup pintu dan buka tutup kunci.

Pada penelitian sebelumnya masih terdapat beberapa kekurangan yaitu belum adanya sistem buka tutup pintu secara otomatis yang di implementasikan pada sebuah mikrokontroller. Pada penelitian ketiga sudah dilakukan implementasi pada sebuah mikrokontroller. Namun, penelitian hanya

diimplementasikan pada pintu mobil dengan berfokus pada sistem pengunciannya saja. Oleh karena itu dibutuhkan sebuah penelitian yang berfokus pada sistem membuka dan menutup pintu sekaligus mengunci dan membuka kunci pintu agar perkembangan teknologi keamanan untuk *smarthome* dapat meningkat pesat.

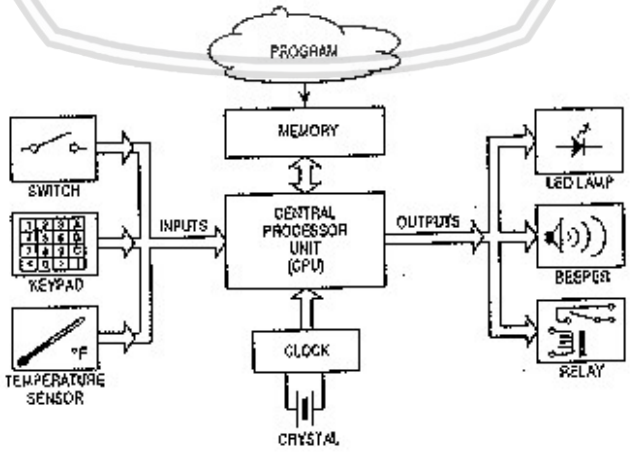
2.2. Dasar Teori

2.2.1 Mikrokontroller

Mikrokontroller adalah suatu IC dengan kepadatan yang sangat tinggi, dimana semua bagian yang diperlukan untuk suatu kontroler sudah dikemas dalam satu keping, biasanya terdiri dari CPU (*Central Processing Unit*), RAM (*Random Access Memory*), EEPROM/EPROM/PROM/ROM, I/O, Serial & Parallel, *Timer*, *Interupt Controller* (Setiawan, 2011). Mikrokontroler adalah sebuah chip yang berfungsi sebagai pengontrol rangkaian elektronik dan umunya dapat menyimpan program didalamnya (Fauzi, et al., 2011).

Berdasarkan definisi yang dikemukakan diatas dapat disimpulkan bahwa mikrokontroller adalah suatu IC yang didesain atau dibentuk dengan kepadatan yang sangat tinggi, dimana semua bagian yang diperlukan suatu kontroler sudah dikemas dalam satu keping, biasanya terdiri dari CPU (*Central Processing Unit*), RAM (*Random Access Memory*), EEPROM/EPROM/PROM/ROM, I/O, Serial & Parallel, *Timer, Interupt Controller* dan berfungsi sebagai pengontrol rangkaian elektronik serta umunya dapat menyimpan program didalamnya.

Seperti umumnya komputer, mikrokontroler adalah alat yang mengerjakan instruksi-instruksi yang diberikan kepadanya. Artinya, bagian terpenting dan utama dari suatu sistem terkomputerisasi adalah program itu sendiri yang dibuat oleh seorang *programmer*. Program ini menginstruksikan komputer untuk melakukan jalinan yang panjang dari aksi-aksi sederhana untuk melakukan tugas yang lebih kompleks yang diinginkan oleh *programmer* (Setiawan, 2011). Pada gambar 2.1 merupakan penjelasan melalui gambar *block hardware* mikrokontroller.

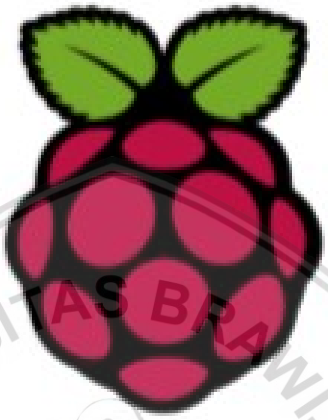


Gambar 2. 1. *Block Hardware* Mikrokontroller

(Sutanto, 1998)

2.2.2 Raspberry Pi

Raspberry Pi adalah sebuah komputer papan tunggal (*single-board computer*) atau SBC berukuran kartu kredit. Raspberry Pi telah dilengkapi dengan semua fungsi layaknya sebuah komputer lengkap, menggunakan SoC (*System-ona-chip*) ARM yang dikemas dan diintegrasikan diatas PCB. Perangkat ini menggunakan kartu SD untuk *booting* dan penyimpanan jangka panjang (Putra, 2012).



Gambar 2. 2 Logo Raspberry Pi

(Raspberry Pi, 2018)

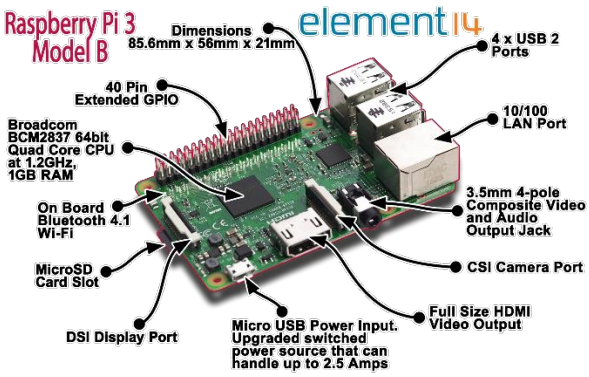
Gambar 2.2 merupakan logo raspberry pi. Raspberry Pi memiliki dua model yaitu model A dan model B. Secara umum raspberry pi model B, 512MB RAM. Perbedaan model A dan B terletak pada memory yang digunakan, Model A menggunakan memory 256 MB dan model B 512 MB. Selain itu model B juga sudah dilengkapai dengan *ethernet port* (kartu jaringan) yang tidak terdapat di model A. Desain raspberry pi didasarkan seputar SoC (*System-on-a-chip*) *Broadcom* BCM2835, yang telah menanamkan prosesor ARM1176JZF-S dengan 700 MHz, *VideoCore* IV GPU, dan 256 Megabyte RAM (model B). Penyimpanan data didesain tidak untuk menggunakan *hard disk* atau *solid-state drive*, melainkan mengandalkan kartu SD (*SD memory card*) untuk *booting* dan penyimpanan jangka panjang. *Hardware* raspberry pi tidak memiliki *real-time clock*, sehingga OS harus memanfaatkan timer jaringan server sebagai pengganti. Namun komputer yang mudah dikembangkan ini dapat ditambahkan dengan fungsi *real-time* (seperti DS1307) dan banyak lainnya, melalui saluran GPIO (*General-purpose input/output*) via antarmuka I^2C (*InterIntegrated Circuit*). Raspberry Pi bersifat *open source* (berbasis Linux), raspberrry pi bisa dimodifikasi sesuai kebutuhan penggunanya. Sistem operasi utama raspberry pi menggunakan Debian GNU/Linux dan bahasa pemrograman *Python*. Salah satu pengembang OS untuk raspberry pi telah meluncurkan sistem operasi yang dinamai Raspbian, Raspbian diklaim mampu memaksimalkan perangkat Raspberry Pi. Sistem operasi tersebut dibuat berbasis Debian yang merupakan salah satu distribusi Linux OS.

2.2.3 Raspberry Pi 3

Raspberry Pi 3 merupakan generasi ketiga dari keluarga Raspberry Pi. Raspberry Pi 3 memiliki RAM 1GB dan grafis *Broadcom VideoCore* IV pada frekuensi clock yang lebih tinggi dari sebelumnya yang berjalan pada 250MHz. Raspberry Pi 3 menggantikan Raspberry Pi 2 model B pada bulan Februari 2016. Kelebihannya dibandingkan dengan Raspberry Pi 2 adalah:

1. A 1.2GHx 64-bit *quad-core* ARMv8 CPU
2. 802.11n *Wireless LAN*
3. *Bluetooth* 4.1
4. *Bluetooth Low Energy* (BLE)
5. 4 *USB port*
6. 40 *pin GPIO*
7. Full HDMI *port*
8. *Port Ethernet*
9. *Combined 3.5mm audio jack*
10. *Composite video*
11. *Camera interface (CSI)*
12. *Display interface* (DSI)
13. Slot kartu Micro SD
14. *VideoCore* IV 3D *graphics core*

Raspberry Pi 3 memiliki bentuk identik dengan Raspberry Pi 2 dan memiliki kompatibilitas lengkap dengan Raspberry Pi 1 dan 2. Raspberry Pi 3 juga direkomendasikan untuk digunakan bagi mereka yang ingin menggunakan Pi dalam proyek-proyek yang membutuhkan daya yang sangat rendah (Man, 2016). Pada Gambar 2.3 merupakan bentuk dan bagian-bagian pada Raspberry Pi 3.



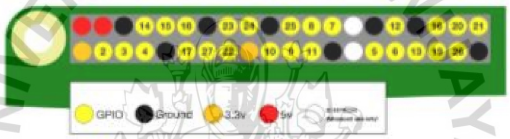
Gambar 2. 3 Tampilan Raspberry Pi 3 Model B

(Man, 2016)

2.2.4 GPIO Raspberry Pi 3

GPIO merupakan sederet pin yang terdiri dari 40 pin dengan berbagai fungsi. Salah satu fitur yang kuat dari Raspberry Pi adalah deretan GPIO (tujuan umum *input / output*) pin di sepanjang tepi atas *pin board*.These adalah antarmuka fisik antara Pi dan dunia luar. Pada tingkat yang paling sederhana, Anda dapat menganggap mereka sebagai *switch* yang Anda dapat mengaktifkan atau menonaktifkan (*input*) atau bahwa Pi dapat mengaktifkan atau menonaktifkan (*output*).

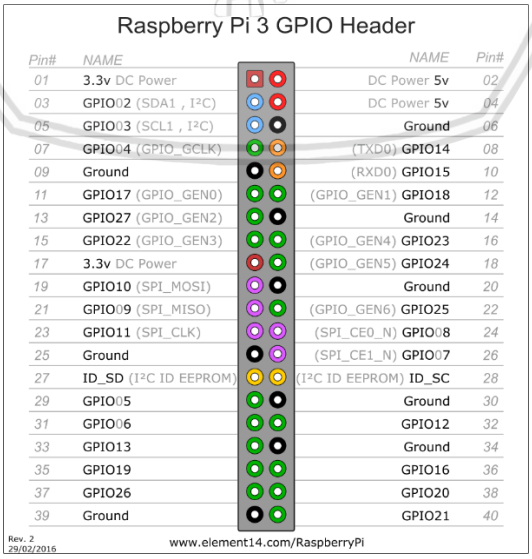
Dari 40 pin, 26 pin GPIO dan yang lain adalah pin *power* atau *ground* (ditambah dua pin ID EEPROM yang tidak harus anda gunakan). Anda dapat memprogram pin untuk berinteraksi dengan cara yang menakjubkan dengan dunia nyata. *Input* tidak harus berasal dari saklar fisik; itu bisa menjadi masukan dari sensor atau sinyal dari komputer lain atau perangkat, misalnya. *output* juga dapat melakukan apa saja, dari menyalakan LED untuk mengirim sinyal atau data ke perangkat lain (Raspberry Pi, 2014). Pada gambar 2.4 merupakan contoh visual GPIO pada Raspberry Pi.



Gambar 2. 4 Raspberry Pi GPIO pin

(Raspberry Pi, 2014)

Penjelasan lebih lanjut mengenai fungsi masing-masing PIN GPIO pada Raspberry Pi khususnya pada Raspberry Pi 3 ada pada gambar 2.5 sebagai berikut:



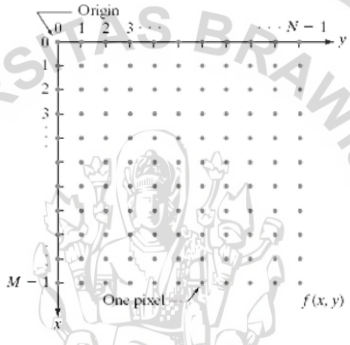
Gambar 2. 5 Raspberry Pi 3 Model B GPIO 40 *Pin Block Pinout*

(Chan, 2015)

2.2.5 Citra Digital

2.2.5.1 Definisi Citra Digital

Secara umum, pengolahan citra digital menunjuk pada pemprosesan gambar 2 dimensi menggunakan komputer. Dalam konteks yang lebih luas, pengolahan citra digital mengacu pada pemprosesan setiap data 2 dimensi. Citra digital merupakan sebuah larik (*array*) yang berisi nilai-nilai real maupun komplek yang direpresentasikan dengan deretan bit tertentu. Suatu citra dapat didefinisikan sebagai fungsi f(*x,y*) berukuran M baris dan N kolom, dengan x dan y adalah koordinat spasial, dan amplitude f di titik koordinat *(x,y*) dinamakan intensitas atau tingkat keabuan dari citra pada titik tersebut. Apabila nilai x,y, dan nilai amplitude f secara keseluruhan berhingga (*finite*) dan bernilai diskrit maka dapat dikatakan bahwa citra tersebut adalah citra digital (Putra, 2010). Gambar 2.6 menunjukan posisi koordinat citra digital.



Gambar 2. 6 Sistem Koordinat Citra Diskrit

(Arief, 2012)

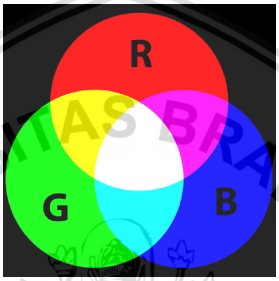
2.2.5.2 Citra Berwarna

Citra berwarna adalah citra yang secara visual memiliki kandungan informasi warna, di mana warna ini direpresentasikan dalam nilai pixel yang mengandung komponen *luminance, hue* dan *chrominance/saturation*. *Luminance* merupakan ukuran dari tingkat kecerahan suatu warna. Menaikan atau menurunkan nilai *luminance* berarti membuat warna tesebut menjadi lebih cerah atau lebih gelap.

Hue adalah sifat utama warna yang direpresentasikan dalam nilai derajat (0°-360°). Warna-warna dasar seperti merah memiliki nilai *hue* 0° atau 360°, hijau memiliki nilai *hue* 120° dan biru memiliki nilai *hue* 240°. Sedangkan untuk warna kuning (campuran warna merah dan hijau) memiliki nilai *hue* 60°, warna oranye (campuran warna kuning dan warna merah) memiliki nilai *hue* 30°, warna ungu (campuran antara warna merah dan biru) memiliki nilai *hue* 300°, warna cyan atau biru langit (campuran antara warna hijau dan biru) memiliki nilai *hue* 180° dan seterusnya warna-warna lain memiliki nilai hue masing-masing (Madenda, 2015).

Chrominance atau *saturation* merepresentasikan tinggi rendahnya cahaya putih dalam sebuah warna. Semakin rendah niulai *chrominance* (mendekati nol)

maka warna tersebut semakin memucat (memutih) hingga menjadi warna putih/abu-abu. Sebaliknya, semakin tinggi nilai *chrominance* maka warna tersebut akan semakin mendekati saturasi atau (*pure colour*). Sebagai contoh, 100% biru adalah warna biru saturasi. Variasi warna ini dapat terbentuk melalui gabungan dari variasi nilai intensitas tiga warna dasar yaitu merah R (*red*), hijau G (*green*), dan biru B (*blue*). Atas dasar inilah citra digital berwarna direpresentasikan dalam computer. Setiap titik atau piksel pada citra berwarna memiliki 3 komponen warna R, G dan B yang masing-masing umumnya dikodekan dalam 8 bit, atau total ketiganya adalah 24 bit (3 byte). Dengan demikian, sebuah citra berwarna dapat memiliki kandungan variasi warna maksimum sebanyak 2^{24} (16.777.216 variasi warna) (Madenda, 2015). Gambar 2.7 merupakan ilustrasi komposisi warna merah R, hijau G dan biru B dalam sebuah gambar.



Gambar 2. 7 Komposisi Warna RGB

(Saputri, 2017)

2.2.5.3 Citra Gray Level

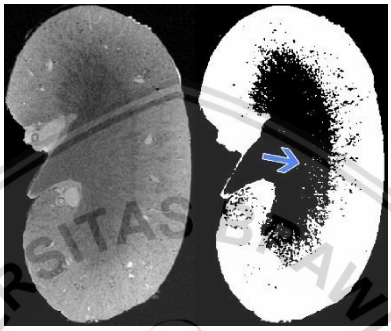
Citra gray-level (skala keabuan) merupakan citra di mana nilai pikselnya hanya diwakilkan oleh nilai *luminance*, yang umumnya dikodekan dalam 8 bit atau artinya memiliki skala keabuan yang bervariasi dari nilai 0 sampai 255 (2^8-1). Nilai 0 merepresentasikan warna hitam dan nilai 255 merepresentasikan warna putih, sedangkan nilai-nilai diantaranya merepresentasikan warna keabuan yang bervariasi dari hitam hingga cerah menuju putih. Citra gray level dapat diperoleh dari citra berwarna melalui transformasi dari ruang warna RGB ke ruang warna lain (HSV, HSL, Lab, YCbCr atau HCL). Komponen Y, V atau L dari ruang warna merepresentasikan citra gray-level yang dimaksud (Madenda, 2015). Gambar 2.8 merupakan contoh dari citra gray level.

Gambar 2. 8 Citra *Grayscale*

(Pamungkas, 2014)

2.2.5.4 Citra Biner

Citra biner merupakan bagian dari citra gray-level yang memiliki dual level keabuan, yaitu 0 untuk warna hitam dan 1 untuk warna putih, sehingga setiap piksel dari citra biner dikodekan dengan hanya menggunakan 1 bit. Citra Biner dihitung dengan menggunakan nilai ambang (*threshold*): Bila nilai piksel lebih kecil daripada batas ambang maka nilai tersebut diubah menjadi 0 hitam, dan bila lebih besar atau sama dengan nilai ambang maka niulai tersebut diubah menjadi 1 putih (Madenda, 2015). Gambar 2.9 memperlihatkan contoh dari thresholding atau mengkonversi dari citra gray-level menjadi citra Biner.



Gambar 2. 9 Contoh *Thresholding*

(Xie, 2013)

2.2.6 *Template Matching*

Template matching adalah adalah proses mencari suatu objek (*template*) pada keseluruhan objek yang berada dalam suatu citra. *Template* dibandingkan dengan keseluruhan objek tersebut dan bila *template* cocok (cukup dekat) dengan suatu objek yang belum diketahui pada citra tersebut maka objek tersebut ditandai sebagai *template*. Perbandingan antara *template* dengan keseluruhan objek pada citra dapat dilakukan dengan menghitung selisih jaraknya, seperti pada persamaan 2.1 dibawah ini (Putra, 2010).

$$D(m,n) = \sum_{j} \sum_{k} [f(j,k) - T(j-m, k-n)]^2 \qquad (\mathbf{2.1})$$

Dengan *f(j,k)* menyatakan citra tempat objek yang akan dibandingkan dengan *template T(j,k)* sedangkan *D(m,n)* menyatakan jarak antara *template* dengan objek pada citra. Pada umumnya ukuran *template* jauh lebih kecil dari ukuran citra. Secara ideal, *template* dikatakan cocok dengan objek pada citra bila *D(m,n) = 0*, namun kondisi seperti ini sulit dipenuhi apalagi bila *template* merupakan suatu citra gray level. Oleh karena itu, aturan yang digunakan untuk menyatakan *template* cocok dengan objek adalah bila memenuhi persamaan 2.2.

$$D(m,n) < L_D\ (m,n) \qquad (\mathbf{2.2})$$

Dengan $L_D\ (m,n)$ merupakan nilai dari threshold.

2.2.7 *Face Detection*

Deteksi wajah dapat dipandang sebagai masalah klasifikasi pola dimana *input*nya adalah suatu citra dan *output*nya adalah label kelas dari citra tersebut. Dalam hal ini terdapat dua label kelas, yaitu wajah dan non-wajah (Sung, 1996). Teknik-teknik pengenalan wajah yang dilakukan selama ini banyak yang menggunakan asumsi bahwa data wajah yang tersedia memiliki ukuran yang sama dan latar belakang yang seragam. Di dunia nyata, asumsi ini tidak selalu berlaku karena wajah dapat muncul di dalam citra dengan berbagai ukuran, berbagai posisi, dan latar belakang yang bervariasi (Hjelm˚as & Low, 2001). Pendeteksian wajah (*face detection*) juga merupakan salah satu tahap awal yang sangat penting sebelum dilakukan proses pengenalan wajah (*face recognition*) (Nugroho & Harjoko, 2004).

2.2.8 *Face Recognition*

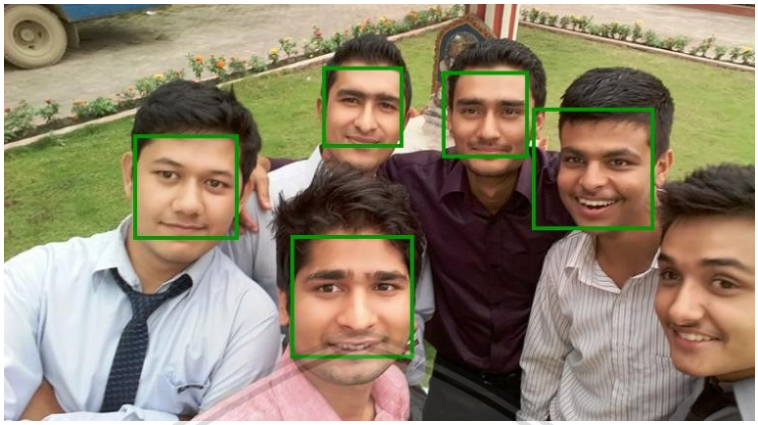
Pengenalan wajah adalah suatu metode pengenalan yang berorientasi pada wajah. Pengenalan ini dapat dibagi menjadi dua bagian yaitu : Dikenali atau tidak dikenali, setelah dilakukan perbandingan dengan pola yang sebelumnya disimpan di dalam database. Metode ini juga harus mampu mengenali objek bukan wajah. Perhitungan model pengenalan wajah memiliki beberapa masalah. Kesulitan muncul ketika wajah direpresentasikan dalam suatu pola yang berisi informasi unik yang membedakan dengan wajah yang lain (Sepritahara, 2012).

2.2.9 *Haar-Like Feature*

Haar like Feature merupakan metode yang lazim digunakan dalam pendeteksian objek. Nama Haar sendiri mengacu pada Haar Wavelet, sebuah fungsi matematika yang berbentuk kotak dan memiliki prinsip seperti pada fungsi fourier (Purwanto, et al., 2015). *Haar-like features* merupakan *rectangular features* (fungsi persegi), yang memberikan indikasi secara spesifik pada sebuah gambar. Prinsip *Haar-like features* adalah mengenali obyek berdasarkan nilai sederhana dari fitur tetapi bukan merupakan nilai piksel dari image obyek tersebut. Metode ini memiliki kelebihan yaitu komputasinya sangat cepat, karena hanya bergantung pada jumlah piksel dalam persegi bukan setiap nilai piksel dari sebuah *image* (Viola & Jones, 2001). Penelitian ini menggunakan pendeteksi wajah yang pertama kali dilakukan oleh Viola dan Jones kemudian dikembangkan oleh Lienhart (Lienhart & Maydt, 2002). Metode yang diusulkan Viola dan Jones menggabungkan empat kunci utama untuk mendeteksi sebuah obyek (Viola & Jones, 2001) :

1. Fitur persegi sederhana, disebut fitur Haar
2. *Integral image* untuk pendeteksian fitur dengan cepat
3. Metode *AdaBoost machine-learning*
4. *Cascade classifier* untuk mengkombinasikan banyak fitur

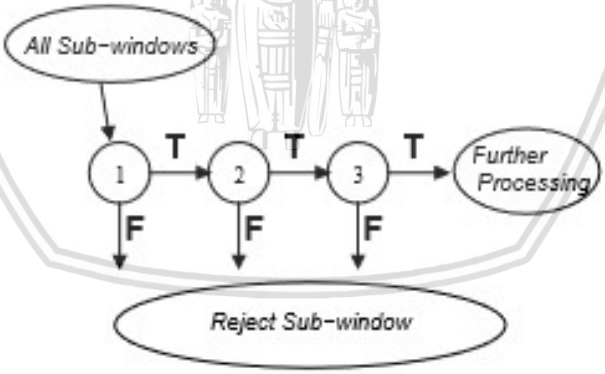
Haar like features memproses citra dalam sebuah kotak persegi dengan ukuran tertentu misalnya 24 x 24 pixel seperti ditunjukkan pada gambar 2.10 sebagai berikut:



Gambar 2. 10 Rectangular *Features Haar Cascade*

(Ghimire, 2017)

Pada kotak hijau diatas terjadi proses filtering atau penyaringan untuk menentukan ada atau tidaknya objek yang akan dideteksi. Proses filterisasi ini akan dilakukan secara kompleks dan bertingkat. Proses Haar yang dilakukan secara massal akan dikenal sebagai *Haar Cascade Classifier* seperti ditunjukkan pada skema filter pada gambar 2.11 sebagai berikut:

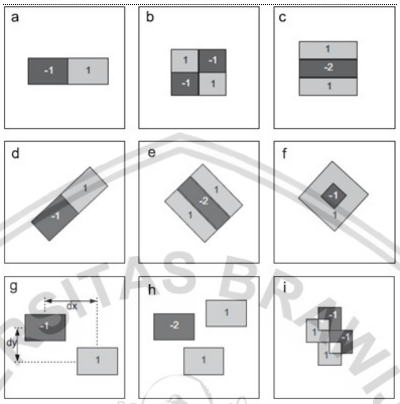


Gambar 2. 11 Skema Pendeteksi Obyek

(Viola & Jones, 2001)

Hasil deteksi dari *Haar-like Feature* sangat tidak akurat jika hanya menggunakan satu fungsi saja. Semakin tinggi tingkatan filter pendeteksian maka semakin tepat pula sebuah obyek dideteksi akan tetapi akan semakin lama proses pendeteksiannya. Pemrosesan *Haar-like features* yang banyak dan terstruktur tersebut diatur dalam *cascade classifier*.

Haar-Wavelet (*Wave* = Gelombang) merupakan gelombang persegi (interval gelap dan interval terang) yang kemudian dibandingkan nilai rata-rata *pixel* keduanya. Apabila perbandingan nilai rata-rata intensitas tersebut berada di atas *threshold* (ambang batas), maka dikatakan memenuhi syarat fitur haar. Untuk gambar bergerak seperti video, proses ini dilakukan secara diskrit dengan mencuplik video pada frame rate tertentu (Pavani, et al., 2010). Ada banyak variasi dari fitur haar, pada gambar 2.12 adalah berbagai macam fitur haar yang dapat digunakan untuk pendeteksian objek:



Gambar 2. 12 Berbagai Variasi Fitur Haar

(Pavani, et al., 2010)

Berikut akan dijelaskan masing-masing fitur pada gambar 2.12 (Pavani, et al., 2010) :

(a), (b) : Fitur Haar yang diusulkan Papageogiou dkk.

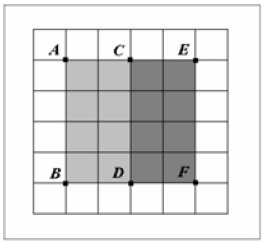
(c) : Fitur Haar yang diusulkan Viola dan Jones.

(d), (e), (f) : Variasi fitur Haar yang diusulkan Leinhardt.

(g), (h) : Penguraian Lietal terhadap *Fitur Haar-like.*

(i) : Fitur Haar-like Viola dan Jones untuk menangkap struktur diagonal dalam penampilan obyek.

Cara menghitung nilai fitur pada gambar 2.12 akan dijelaskan oleh gambar 2.13 dimana contoh perhitungan menggunakan fitur haar *(a)* :



Gambar 2. 13 Fitur Persegi *Haar-like*

(Sajati, 2015)

$$(\boldsymbol{AEBF}) = (\boldsymbol{ACBD}) - (\boldsymbol{CEDF}) \quad \textbf{(2.3)}$$

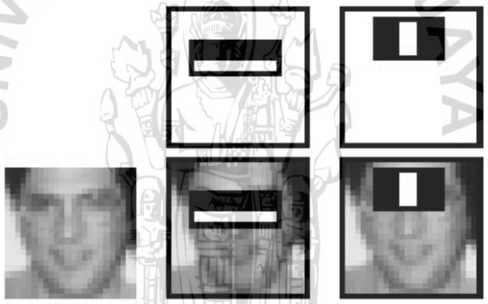
(AEBF) = Nilai Fitur

(ACBD) = Jumlah nilai setiap piksel daerah *(ACBD)*

(CEDF) = Jumlah nilai setiap piksel *(CEDF)*

Apabila Nilai Fitur *(AEBF)* memiliki nilai di atas *threshold* maka dikatakan memenuhi syarat. Seperti dijelaskan pada gambar 2.13. , apabila sebuah fitur dikatakan tidak memenuhi syarat, maka area *(AEBF)* tidak terdapat obyek yang dideteksi dan area perseginya berpindah lokasi akan tetapi jika persegi (AEBF) memenuhi fitur, maka aturan fitur berikutnya dilakukan. Jika semua syarat fitur dipenuhi dikatakan pada persegi ABFE dikatakan terdapat obyek.

Proses pendeteksian objek menggunakan fitur haar akan diilustrasikan pada gambar 2.14. Objek yang akan dideteksi ialah wajah melalui tampak depan :



Gambar 2. 14 Pendeteksian Wajah Menggunakan *Haar-like Features*

(Viola & Jones, 2001)

2.2.10 *Integral image*

Sebuah citra digital memiliki komponen nilai RGB (kombinasi dari warna merah, hijau dan biru). Dari nilai RGB tersebut dapat diketahui nilai *grayscale* (derajat keabu-abuan) yang dihitung menggunakan persamaan 2.4 sebagai berikut:

$$Grayscale_{pixel} = 0.2989\ R + 0.5870\ G + 0.1140\ B \quad \textbf{(2.4)}$$

(R) : Red (Merah), skala warna merah pada piksel

(G) : Green (Hijau), skala warna merah pada piksel

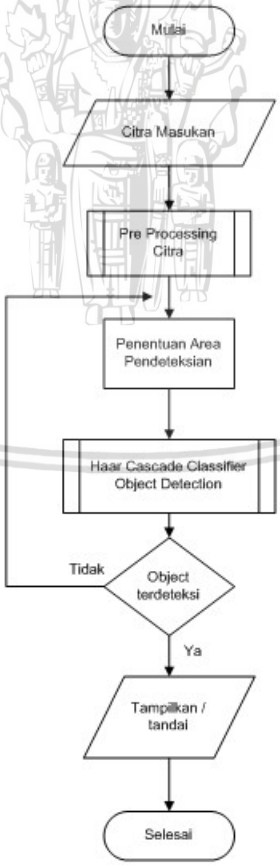
(B) : Blue (Biru), skala warna merah pada piksel

Sebagai contoh sebuah *pixel* memiliki kombinasi warna *R* = 100, *B* = 100 dan *G* = 100 maka nilai *grayscale* menurut persamaan 2.4 di atas sama dengan 99.99. Sebuah contoh citra yang diubah dari citra rgb menjadi *grayscale* ditunjukkan pada gambar 2.15 sebagai berikut:

Gambar 2. 15 Perbedaan Citra Asli dengan *Grayscale*

(Mahesh, 2016)

Pendeteksian objek menggunakan metode haar melalui beberapa proses yang runtut dimulai dari input berupa citra hingga objek berhasil dideteksi. Gambar 2.16 akan menjelaskan proses pendeteksian menggunakan diagram alir/ *flowchart* agar lebih mudah dimengerti :



Gambar 2. 16 Pendeteksian Obyek dengan *Haar Cascade Clasifier*

(Sajati, 2015)

Citra integral merupakan sebuah citra yang nilai tiap piksel-nya merupakan penjumlahan nilai piksel atas dan kirinya. Penggunaan citra integral merupakan cara paling efektif untuk mengetahui jumlah nilai tiap daerah piksel yang ingin diketahui nilainya tanpa harus melakukan *scanning* piksel berulang-ulang. Sebagai contoh pada gambar 2.17 sebuah daerah persegi yang akan di-*scan* menggunakan persegi gelap terang memiliki nilai sebagai berikut:

2	3	1	3	6	5
3	1	2	5	4	4
1	2	3	4	5	4
4	4	5	6	7	3
5	4	3	6	7	4
1	2	3	4	5	6

Gambar 2. 17 Citra Masukan

(Sajati, 2015)

Gambar 2.17 merupakan nilai dari setiap piksel yang merupakan nilai *grayscale* yang akan dicari citra integralnya menggunakan salah satu fitur haar seperti yang terlihat pada gambar 2.18 sebagai berikut:

2	3	1	3	6	5
3	1	2	5	4	4
1	2	3	4	5	4
4	4	5	6	7	3
5	4	3	6	7	4
1	2	3	4	5	6

Gambar 2. 18 Persegi Fitur Haar Pada Citra Masukan

(Sajati, 2015)

Proses perhitungan selisih nilai gelap dan nilai terang pada persamaan 2.3 maka akan menghasilkan nilai fitur haar sama dengan 22. Cara perhitungan tersebut memang cukup efektif jika hanya menggunakan satu fitur saja. Namun akan berbeda jika kita harus menggunakan ratusan fitur dengan skala yang berbeda-beda. Untuk itu, penggunaan citra integral akan lebih efektif. Berikut ialah cara menghitung fitur nilai haar menggunakan *Summed Area Table* atau yang dikenal sebagai *Integral image*.

Proses *integral image* matriks citra ditentukan menggunakan persamaan 2.5 sebagai berikut:

$$s(x,y) = i(x,y) + s(x-1,y) + s(x,y-1) - s(x-1,y-1) \quad \textbf{(2.5)}$$

s(x,y) : nilai citra integral pada bidang *(x,y).*

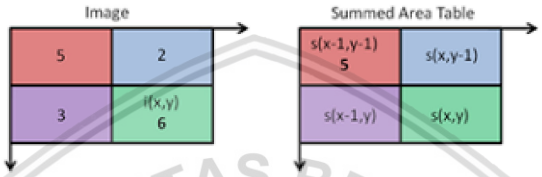
i(x,y) : intensitas/nilai piksel citra asli.

s(x-1,y) : merupakan nilai citra integral dari piksel tetangga sebelah kiri.

s(x,y-1) : merupakan nilai citra integral dari piksel tetangga sebelah atas.

s(x-1,y-1) : merupakan nilai citra integral diagonal dari *s(x,y).*

Agar lebih mudah untuk dimengerti, pada gambar 2.19 merupakan ilustrasi dari persamaan 2.5 yang dituangkan dalam bidang (x,y):



Gambar 2. 19 Pixel Tetangga pada Proses Integral image

(Hendry, 2012)

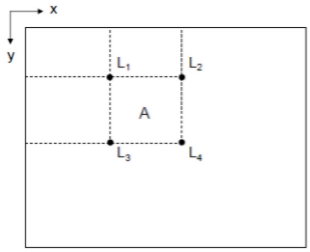
Dengan menggunakan persamaan 2.5 , *integral image* dari citra masukan gambar 2.20 sebagai berikut:

2	5	6	9	15	20
5	9	12	20	30	39
6	12	18	30	45	58
10	20	31	49	71	87
15	29	43	67	96	116
16	32	49	77	111	137

Gambar 2. 20 Matriks Integral Image dari Citra Masukan

(Sajati, 2015)

Proses selanjutnya ialah menghitung jumlah nilai tiap piksel pada suatu daerah yang ditandai. Gambar 2.21 merupakan ilustrasi bagian pada citra yang akan dicari jumlah nilai tiap pikselnya :



Gambar 2. 21 Luas Daerah Nilai Piksel Yang Akan Dihitung

(Derpanis, et al., 2007)

$$A = L_1 + L_4 - (L_2 + L_3) \quad \textbf{(2.6)}$$

(A) = Luas daerah pada citra yang ingin diketahui jumlah nilai pikselnya.

(L_1) = Nilai *integral image* sebelah diagonal dari piksel pojok kiri atas.

(L_2) = Nilai *integral image* sebelah atas dari piksel pojok kanan atas.

(L_3) = Nilai *integral image* sebelah kiri dari piksel pojok kiri abawah.

(L_4) = Nilai *integral image* dari piksel pojok kanan bawah.

Sehingga jika menggunakan persamaan 2.3 maka nilai fitur haar pada gambar 2.17 ialah sebagai berikut :

$$Nilai\ Fitur = \big(32 + 2 - (16 + 5)\big) + \big(77 + 6 - (49 + 9)\big) - \big(49 + 5 - (32 + 6)\big)$$

$$Nilai\ Fitur = 13 + 25 - 16$$

$$Nilai\ Fitur = 22$$

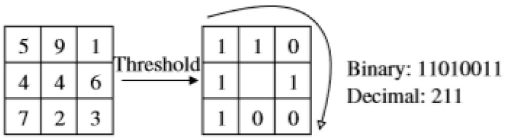
Hasil ini sesuai dengan perhitungan secara manual menggunakan persamaan 2.3. Nilai 22 tersebut kemudian dibandingkan dengan *threshold* yang sudah ditentukan sebagai pendeteksian obyek. Apabila nilai fitur Haar lebih tinggi daripada *threshold*, maka dapat dikatakan pada area tersebut memenuhi filter Haar. Sesuai diagram alir pada gambar 2.17 , proses ini akan dilanjutkan untuk menguji kembali area tersebut dengan filter Haar yang lain dan apabila seluruh filter Haar terpenuhi maka dikatakan pada area tersebut terdapat objek yang diamati. Pada penelitian ini dilakukan pendeteksian objek berupa wajah tampak depan.

2.2.11 *Local binary Pattern*

Operator *Local Binary Patern* atau sering dikenal dengan sebutan LBP adalah salah satu deskriptor tekstur terbaik dan telah banyak digunakan dalam berbagai aplikasi. LBP telah terbukti sangat diskriminatif, yaitu variasi untuk perubahan tingkat abu-abu monoton dan efisiensi komputasi, membuatnya cocok untuk tugas deskripsi untuk gambar yang menuntut analisis. Ide menggunakan LBP untuk deskripsi wajah didukung oleh fakta wajah dapat dilihat sebagai komposisi pola mikro yang dapat dijelaskan oleh sebuah operator.

Local Binary Pattern (LBP) didefinisikan sebagai ukuran tekstur *gray-scale* invarian, berasal dari definisi umum tekstur di daerah sekitar. Operator LBP dapat dilihat sebagai pendekatan kesatuan dengan model statistik dan struktur tradisional berbeda dari analisis tekstur. Secara sederhana, LBP adalah sebuah

kode biner yang menggambarkan pola tekstur lokal. Hal ini dibangun dengan lingkungan batas dengan nilai abu-abu dari pusatnya (Ahonen, et al., 2004). Pada gambar 2.22 berikut merupakan contoh standar operator LBP pada piksel berukuran 3 x 3 :



Gambar 2. 22 Operator Standar LBP

(Ahonen, et al., 2004)

Setiap piksel memiliki nilai hasil *grayscale*, kemudian dilakukan *threshold* berpusat pada titik tengah. Piksel yang memiliki nilai sama atau lebih dibandingkan dengan titik tengah diberi nilai 1 selain itu diberi nilai 0. Kemudian nilai LBP didapat dari penjumlahan dua pangkat nilai angka yang bernilai satu, pada persamaan persamaan 2.8 berikut akan dijabarkan bagaimana nilai diatas didapat:

$$LBP_{P.R} = \sum_{p=0}^{p-1} s\,(g_p - g_c)\,2^p \qquad s\,(x) = \begin{cases} 1, if\ x \geq 0; \\ 0, otherwise. \end{cases} \qquad \textbf{(2.8)}$$

$(LBP_{P.R})$ = Nilai *Local Binary Pattern*

(P) = Jumlah piksel yang ada di sekitar titik pusat

(R) = Radius/jarak piksel sekitar dengan titik pusat

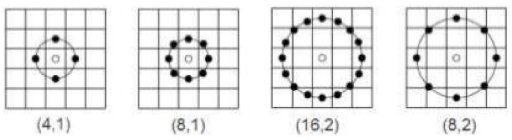
(g_P) = Nilai piksel sekitar

(g_c) = Nilai piksel pusat

$\big(S(x)\big)$ = Merupakan nilai hasil selisih nilai piksel sekitar dan piksel tengah

(x) = Selisih nilai piksel sekitar dan piksel tengah

Operator pada LBP memiliki label yang ditandai dengan P dan R. P mewakili jumlah pixel tetangga yang digunakan dalam komputasi sementara R adalah radius atau jarak antara pixel titik pusat dan pixel tetangga (Biglari, et al., 2014).

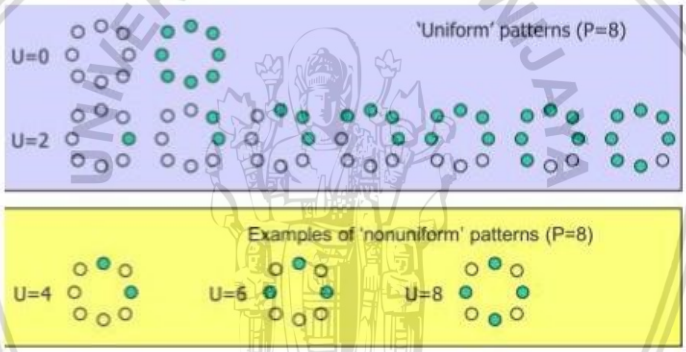


Gambar 2. 23 Varian LBP

(Biglari, et al., 2014)

Dalam aplikasi analisis tekstur banyak diinginkan untuk memiliki fitur yang invarian atau kuat untuk rotasi gambar input. Sebagai LBP, pola P,R diperoleh dari sampel yang diambil disekitar piksel pusat, rotasi gambar input memiliki dua efek: setiap lingkungan lokal diputar ke lokasi pixel lainnya, dan dalam masing-masing lingkungan, titik sampling pada lingkaran yang mengelilingi titik pusat diputar ke orientasi yang berbeda.

Perpanjangan pada operator aslinya menggunakan *Uniform Pattern.* Untuk ini, ukuran keseragaman pola yang digunakan: U ("pola") adalah jumlah bitwise transisi dari 0 ke 1 atau sebaliknya ketika pola bit dianggap melingkar. Pola biner lokal disebut seragam jika ukuran keseragaman adalah 2. Misalnya, pola 00000000 (0 transisi), 01110000 (2 transisi) dan 11001111 (2 transisi) adalah seragam sedangkan pola 11001001 (4 transisi) dan 01010011 (6 transisi) tidak. Dalam pemetaan LBP seragam ada yang terpisah output label untuk setiap pola seragam dan semua non-seragam pola ditugaskan ke label tunggal. Dengan demikian, jumlah label output yang berbeda untuk pemetaan pola-pola (Pietikainen, et al., 2011).



Gambar 2. 24 Uniform patterns

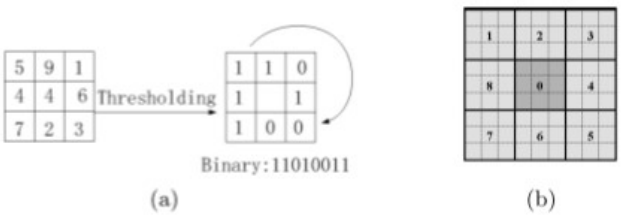
(Pietikainen, et al., 2011)

2.2.11.1 *Local Binary Pattern* untuk *Detection*

Tujuan deteksi adalah untuk menentukan lokasi dan besarnya wajah manusia pada gambar atau citra. Pada proses *local binary pattern detection* yang diterapkan OpenCV ada dua tahapan penting dalam mendeteksi wajah yaitu *Multi-scale Block Local Binary Pattern* dan *Gentle AdaBoost*.

2.2.11.2 *Multi-scale Block Local Binary Pattern*

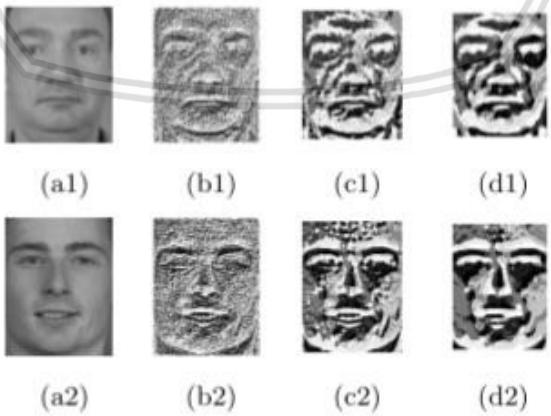
Operator LBP asli label pixel dari suatu gambar dengan thresholding tetangga 3x3 masing-masing pixel nilai pusat dan mempertimbangkan hasil sebagai string biner atau angka desimal. Kemudian histogram dari label dapat digunakan sebagai deskriptor tekstur. Sebuah ilustrasi dari operator LBP dasar ditampilkan pada gambar 2.30 Multiscale LBP merupakan pengembangan dari LBP, yang berhubungan langsung dengan penggunaan pixel tetangga (Liao, et al., 2007).



Gambar 2. 25 (a) Operator Dasar LBP (b) Contoh MB-LBP

(Liao, et al., 2007)

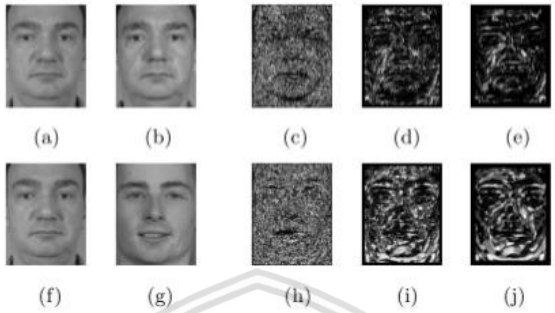
Operator dalam MB-LBP, operator perbandingan antara pixel tunggal dalam LBP hanya diganti dengan perbandingan antara nilai-nilai abu-abu rata-rata pada sebagian daerah. Setiap wilayah sub blok persegi yang mengandung pixel tetangga. Keseluruhan filter terdiri dari 9 blok. kita mengambil ukuran s sebagai parameter, dan S X S yang menunjukkan skala operator MB-LBP (khususnya, 3 x 3 MB-LBP sebenarnya adalah LBP asli). Perhatikan bahwa nilai-nilai skalar dari rata-rata selama blok dapat dihitung sangat efisien dari tabel. Untuk alasan ini, MB-LBP ekstraksi fitur juga bisa sangat cepat karena hanya menimbulkan biaya lebih sedikit dari 3 x 3 LBP operator. Pada gambar 2.26 diberikan contoh MB-LBP gambar wajah disaring oleh 3 x 3, 9 x 9 dan 15 x 15 blok. Dari contoh ini kita bisa melihat pengaruh parameter s. Untuk skala kecil, lokal, pola mikro dari struktur wajah juga terwakili, yang bermanfaat untuk membedakan rincian lokal wajah. Di sisi lain, menggunakan daerah rata-rata nilai daerah, filter skala besar mengurangi kebisingan, dan membuat representasi yang lebih kuat, dan informasi skala besar memberikan informasi melengkapi rincian skala kecil. Tapi banyak informasi diskriminatif juga turun. Biasanya, filter dari berbagai skala harus hati-hati dipilih dan kemudian disatukan untuk mencapai kinerja yang lebih baik.



Gambar 2. 26 Perbedaan citra wajah menggunakan operator MB-LBP (a) gambar awal (b) 3x3 MB-LBP (c) 9x9 MB-LBP (d) 15x15 MB-LBP

(Liao, et al., 2007)

Selanjutnya, pada gambar 2.27 menunjukkan MB-LBP fitur untuk gambar perbedaan intra personal dan ekstra-personal (piksel yang terang menunjukkan perbedaan yang lebih besar). Perbedaan gambar digunakan untuk menunjukkan kekuatan diskriminatif dari MB-LBP.



Gambar 2. 27 Perbedaan citra pada intra dan extra personal image (a)(b)gambar intra-personal image (c)(d)(e)hasil gambar dari 3x3,9x9,15x15 MB-LBP (f)(g)gambar extra-personal image (h)(i)(j)hasil gambar dari 3x3,9x9,15x15 MB-LBP

(Liao, et al., 2007)

2.2.11.3 *Gentle AdaBoost*

Gentle AdaBoost merupakan pengembangan dari metode *AdaBoost. Gentle AdaBoost* menggunakan regresi pangkat terkecil untuk meminimasi fungsi (Ferreira, 2007).

Input:
• Training set $Z=\{z_1,z_2,\ldots,z_N\}$, with $z_i=(x_i,y_i)$.
• M, the maximum number of classifiers.
Output:
• H(x), a classifier suited for the training set.

Step 1
Initialize the weights $w_i = 1/N$, i={1,...,N}

Step 2
for m=1 to M
a) Train $H_m(x)$ by weighted least squares of x_i to y_i, with weights w_i
b) Set $H(x) = H(x) + H_m(x)$
c) Set weights $w_i \leftarrow w_i \exp\left(-y_i H_m(x)\right)$ and renormalize.

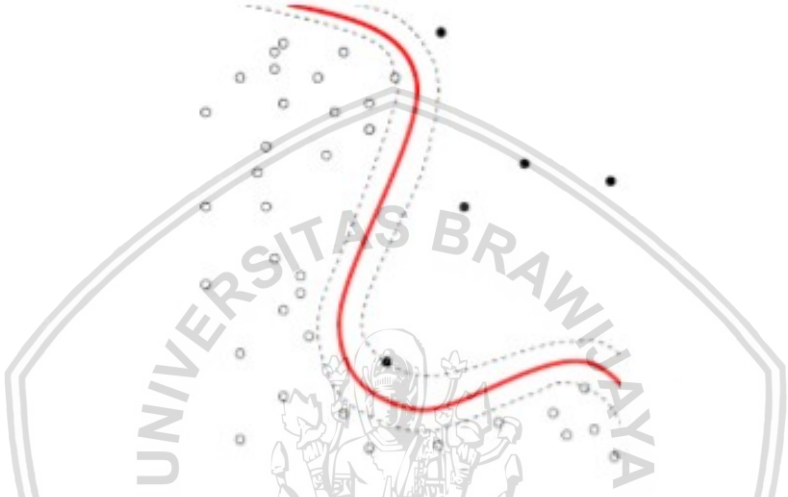
Step 3
Produce the final classifier.

$$H(x) = \operatorname{sgn}\left(\sum_{n=1}^{M} H_n(x)\right)$$

Gambar 2. 28 Algoritma *Gentle AdaBoost*

(Ferreira, 2007)

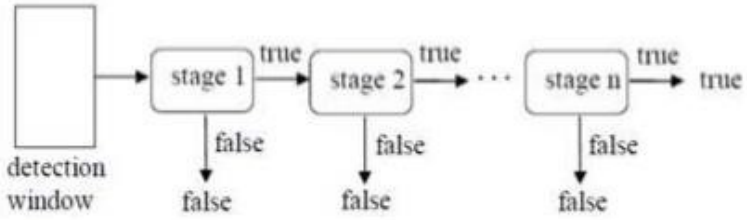
Pada gambar 2.28, fungsi *H* dicari dengan menentukan kuadrat terkecil *y* dan *x* dengan nilai *weight w*. Kemudian dilakukan perulangan pada fungsi yang didapat dengan fungsi baru. Penambahan kedua fungsi menghasilkan fungsi baru. Setelah didapat fungsi yang baru dilakukan normalisasi. Tujuan *gentle adaboost* adalah menentukan suatu fungsi yang mewakili seluruh hasil data training yang telah dimasukkan. Kemudian dari hasil fungsi tersebut ditentukan batas error yang digunakan sebagai tolak ukur data *input* baru. Gambar 2.29 menggambarkan fungsi garis yang terbentuk dari *Adaboost.*



Gambar 2. 29 Fungsi garis terbentuk dari *AdaBoost*

(Ferreira, 2007)

Klasifikasi menggunakan *AdaBoost* dilakukan dalam beberapa tahap. Tahapan tersebut dilakukan dengan menetapkan penambahan jumlah bin setiap kenaikan tahap klasifikasi data citra wajah. Pada gambar 2.30, klasifikasi dilakukan bertahap dengan tujuan menghemat waktu komputasi. Penghematan waktu komputasi terjadi pada klasifikasi kasar pada tahap awal. Sehingga tidak perlu dilakukan klasifikasi detail atau menggunakan semua bin pada semua data citra (Ferreira, 2007).

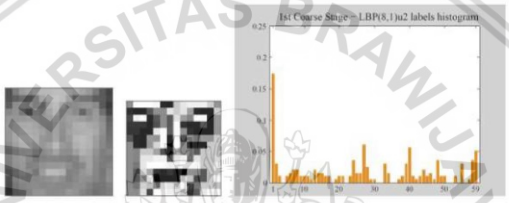


Gambar 2. 30 Proses klasifikasi *AdaBoost*

(Ferreira, 2007)

2.2.11.4 *Local Binary Pattern Histogram* untuk *Recognition*

Pada proses LBP, dibentuk histogram dengan menambahkan nilai setiap blok sesuai dengan pola biner yang sama. Pola biner tersebut yang digunakan berdasakan pattern uniform yang digunakan. Pada prakteknya suatu image dibagi menjadi beberapa bagian/blok. Hal ini dilakukan untuk mempercepat proses komputasi yang dilakukan. Pada Gambar 2.31, proses ekstrasi wajah dilakukan dengan menggunakan local binary pattern. Uniform pattern yang digunakan adalah 2. Alasan digunakan pengambilan pola biner hanya pada U2 karena pola biner U2 mewakili sekitar 80% dari jumalh pixel yang ada (Lopez, 2010). Sehingga hal ini tidak mengganggu proses identifikasi wajah. Dari 8 biner terdapat 256 pola biner yang di dapat dari 2 pangkat 8. Namun pola biner *uniform pattern* 2 hanya terdapat 58 pola biner uniform. Tapi perlu ditambah kan 1 lagi sebagai non-uniform sebagai perwakilan dari semua pola nonuniform. Sehingga jumlah bin yang digunakan adalah 59 bin.



(a) **(b)** **(c)**

Gambar 2. 31 Proses penentuan nilai pada LBP a) Contoh gambar wajah b) hasil LBP c) Histogram LBP uniform pattern 2

(Lopez, 2010)

2.2.12 *Euclidean Distance*

Euclidean distance adalah matrik yang paling sering digunakan untuk menghitung kesamaan 2 vektor. Euclidean distance menghitung akar dari kuadrat perbedaan 2 vektor (Wurdianarto, et al., 2014). Rumus dari euclidian distance ialah sebagai berikut :

$$d_{ij} = \sqrt{\sum_{k=1}^{n}(X_{ik} - X_{jk})^2} \quad \textbf{(2.9)}$$

Keterangan :

d_{ij} = tingkat perbedaan

n = jumlah vektor

x_{ik} = vektor citra masukan

x_{jk} = vektor citra pembanding/keluaran

Untuk membuktikan rumus pada persamaan 2.9, berikut merupakan contoh sederhana penggunaan rumus :

Terdapat 2 vektor ciri berikut:

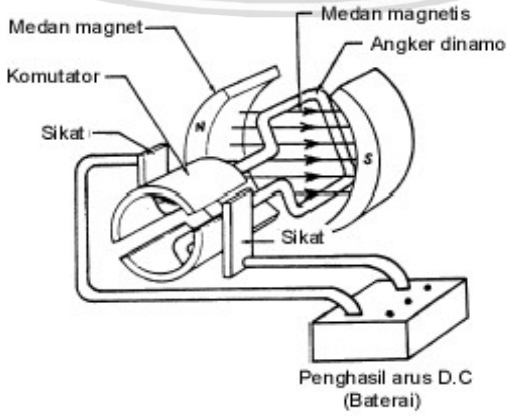
$$A = [0, 3, 4, 5]$$
$$B = [7, 6, 3, -1]$$

Euclidean distance dari vektor A dan B adalah sebagai berikut :

$$d_{AB} = \sqrt{(0-7)^2 + (3-6)^2 + (4-3)^2 + (5-(-1))^2}$$
$$= \sqrt{49 + 9 + 1 + 36} = = 9.747$$

2.2.13 Motor DC

Motor DC memerlukan suplai tegangan yang searah pada kumparan medan untuk diubah menjadi energi mekanik. Kumparan medan pada motor dc disebut stator (bagian yang tidak berputar) dan kumparan jangkar disebut rotor (bagian yang berputar). Jika terjadi putaran pada kumparan jangkar dalam pada medan magnet, maka akan timbul tegangan (GGL) yang berubah-ubah arah pada setiap setengah putaran, sehingga merupakan tegangan bolak-balik. Prinsip kerja dari arus searah adalah membalik phasa tegangan dari gelombang yang mempunyai nilai positif dengan menggunakan komutator, dengan demikian arus yang berbalik arah dengan kumparan jangkar yang berputar dalam medan magnet. Bentuk motor paling sederhana memiliki kumparan satu lilitan yang bisa berputar bebas di antara kutub-kutub magnet permanen (Zuhal, 1990). Gambar 2.32 menunjukan ilustrasi dari motor D.C Sederhana.



Gambar 2. 32 Motor D.C Sederhana

(Zuhal, 1990)

Catu tegangan dc dari baterai menuju ke lilitan melalui sikat yang menyentuh komutator, dua segmen yang terhubung dengan dua ujung lilitan. Kumparan satu lilitan pada gambar di atas disebut angker dinamo. Angker dinamo adalah sebutan untuk komponen yang berputar di antara medan magnet.

2.2.14 Python

Python merupakan salah satu bahasa pemrograman yang populer di dunia kerja Indonesia. Selain itu di ranah akademik pun banyak akademisi yang menggunakan Python untuk menyelesaikan penelitiannya di bidang komputasi sains, robotika, *data science*, ekonomi, antariksa dan berbagai macam bidang lainnya. Python secara *default* telah terpasang di beberapa sistem operasi berbasis Linux seperti Ubuntu, Linux Mint, dan Fedora. Untuk sistem operasi lain, sudah tersedia *installer* yang disediakan untuk sistem operasi tersebut. Banyak hal yang dapat Anda jelajahi ketika menggunakan bahasa pemrograman Python. Beberapa *package* Python yang populer di Python antara lain:

1. Django, *web framework*
2. Scipy dan Scikit, pustaka untuk membuat aplikasi *machine learning* dan *artificial intelligence*
3. Tornado, pustaka untuk membuat aplikasi *web*, *websocket*, dan *asynchronous programming*
4. Celery, pustaka untuk membuat *asynchronous task*
5. OpenCV Python, pustaka untuk membuat aplikasi *computer vision*
6. Matplotlib, pustaka untuk membuat grafik untuk keperluan saintifik

Selain itu Python pun memiliki sebuah *package manager* yang populer dan unggul yang dinamakan dengan PIP. Dengan menggunakan PIP, Anda dapat mulai memasang atau menghapus pustaka Python yang akan atau tidak digunakan lagi. Di Indonesia sendiri banyak sekali *website* yang sudah ditenagai dengan menggunakan Python. Beberapa *website* yang ditenagai Python antara lain: KelasKita, CodeSaya, Kargo.Co.Id, dan banyak lainnya Pada gambar 2.33 adalah logo dari bahasa pemrograman python.

Gambar 2. 33 Logo Python

(Python, n.d.)

2.2.15 *Webcamera*

Camera Web (WebCam) merupakan kamera video digital kecil yang disambungkan ke computer melalui port USB ataupun port COM. Webcam memiliki fungsi untuk memudahkan kita dalam melakukan pesan cepat seperti *chat* melalui *video* ataupun bertatap muka secara langung (*video call*). *Web cam* juga memiliki fungsi lain yaitu mentransfer sebuah media secara *live*, namun perlu kalian ketahui bahwa kebanyakan pengguna menggunakan ini hanya untuk *chat video/video call* (Ali, 2017).

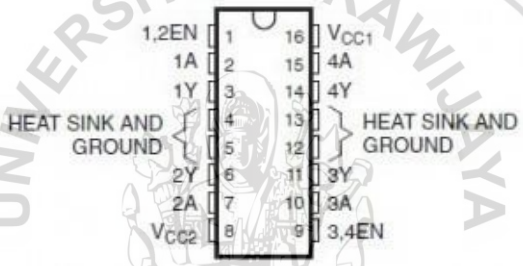
Satu unit *Webcam* sederhana biasanya terdiri dari satu lensa standar, dipasang di papan sirkuit untuk bisa menangkap sinyal gambar, termasuk casing depan,casing samping yang berfungsi untuk menutupi lensa standard dan mempunyai sebuah lubang di casing depan yang berfungsi untuk memasukan gambar. Kabel *support* yang di ciptakan dari bahan yang fleksibel ini, satu ujungnya di hubungkan dengan kamera dan satu ujungnya lagi diberikan penghubung atau *connector*, kabel ini di kontrol untuk menyesuaikan kesemuanya termasuk ketinggian, arah sudut pandang yang ada pada *webcam*. Setiap *webcam* biasanya di lengkapi dengan *software* yang akan membantu mengambil gambar -gambar dari kamera secara terus menerus atau secara interval (waktu tertentu) dan menyebarkannya melalui koneksi internet yang ada. Ada berbagai macam metode penyiaran metode yang paling sering di gunakan adalah *hardware* yang mengubah gambar ke bentuk file JPG dan menguoploadnya ke *web server* memakai *file transfer protocol* (FTP). Didalam menggunakan webcam ini di butuhkan koneksi internet yang relative kencang agar semuanya bisa berjalan dengan lancar. Kalian juga bisa menghindari penggunaan kabel yang ada dengan menggunakan koneksi *Ethernet* ataupun WIFI. Gambar 2.34 merupakan contoh dari *webcam.*

Gambar 2. 34 *Webcam*

(Agung, 2016)

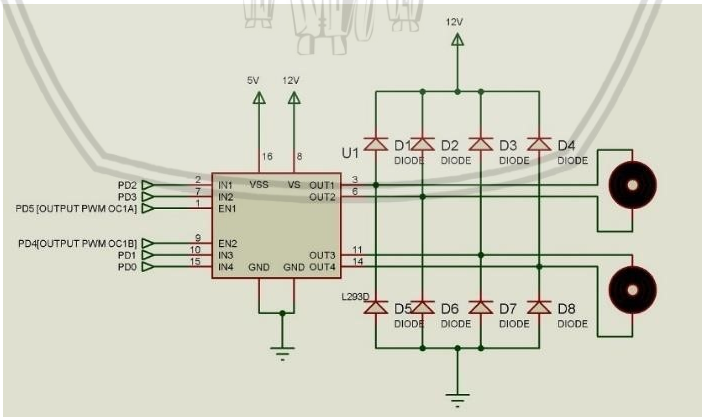
2.2.16 IC L293D

IC L293D adalah *IC* yang didesain khusus sebagai driver motor DC dan dapat dikendalikan dengan rangkaian TTL maupun mikrokontroller. Motor DC yang dikontrol dengan driver *IC L293D* dapat dihubungkan ke *ground* maupun ke sumber tegangan positif karena di dalam *driver L293D* system driver yang digunakan adalah *system totem pool*. Dalam 1 unit *chip IC L293D* terdiri dari 4 buah driver motor DC yang berdiri sendiri dengan kemampuan mengalirkan arus 1 Ampere tiap drivernya. Sehingga dapat digunakan untuk membuat *driver Hbridge* untuk 2 buah motor DC. *IC L293D* terdiri dari 16 pin dan hadir dalam dua versi, yaitu L293D dan L293, huruf D menunjukkan adanya dioda yang berfungsi untuk mengurangi induksi tegangan, jadi motor yang digunakan jadi lebih aman dan awet. Pada gambar 2.35 dan 2.36 akan ditunjukan kontruksi pin driver motor DC IC L293D. Prinsip kerja driver menggunakan L293D, yaitu dengan memberikan tegangan 5 volt sebagai VCC pada pin 16 dan 12 volt pada pin 8 untuk tegangan motor, maka IC siap digunakan (Syahrul, 2014).



Gambar 2. 35 Konstruksi Pin Driver Motor DC IC L293D

(Anon., 2012)



Gambar 2. 36 Konfigurasi Driver Motor DC Menggunakan IC L293D

(Munandar, 2013)

Pin EN1 adalah pin untuk menghidupkan motor 1 (ON/OFF) biasanya pin EN1 dihubungkan dengan PWM untuk mengontrol kecepatan motor. Sementara untuk EN2 fungsinya sama dengan EN1 bedanya EN2 untuk mengontrol motor DC 2. Sementara untuk mengontrol arah putar motor dapat dilihat pada tabel 2.1.

Tabel 2. 1 Arah Putar Motor DC

IN 1	IN 2	KONDISI MOTOR DC 1
0	0	STOP
1	0	PUTAR BERLAWANAN JARUM JAM (CCW)
0	1	PUTAR SEARAH JARUM JAM (CW)
1	1	STOP

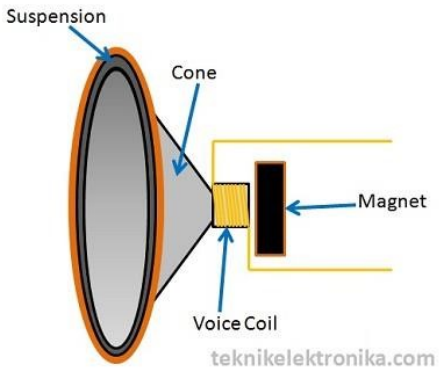
IN 3	IN 4	KONDISI MOTOR DC 2
0	0	STOP
1	0	PUTAR BERLAWANAN JARUM JAM (CCW)
0	1	PUTAR SEARAH JARUM JAM (CW)
1	1	STOP

Jika IN1 diberi *logic* 1 dan IN2 diberi *logic* 0, maka motor A akan berputar kebalikan arah jarum jam. Dan sebaliknya jika IN1 diberi *logic* 0 dan IN2 diberi *logic* 1, maka motor 1 akan berputar searah jarum jam. Jika memberi logik 1 atau 0 pada IN1 dan IN2 bersamaan, Motor 1 akan berhenti (Pengereman Secara Cepat). Begitu juga dengan motor 2. Sementara untuk mengatur kecepatan motor adalah dengan mengatur input dari *enable* 1 (pin1) dan *enable* 2 (pin9) menggunakan *PWM* (*Pulse Width Modulation*) (Munandar, 2013).

2.2.17 *Speaker*

Kita dapat mendengarkan musik radio, mendengarkan suara dari drama televisi ataupun suara dari lawan bicara kita di ponsel, semua ini karena adanya komponen Elektronika yang bernama *Loudspeaker* yang dalam bahasa Indonesia disebut dengan Pengeras Suara. *Loudspeaker* atau lebih sering disingkat dengan Speaker adalah Transduser yang dapat mengubah sinyal listrik menjadi Frekuensi *Audio* (sinyal suara) yang dapat didengar oleh telinga manusia dengan cara mengetarkan komponen membran pada *Speaker* tersebut sehingga terjadilah gelombang suara.

Sebelum kita membahas lebih lanjut mengenai *Loadspeaker* (Pengeras Suara), sebaiknya kita mengetahui bagaimana suara dapat dihasilkan. Yang dimaksud dengan "Suara" sebenarnya adalah Frekuensi yang dapat didengar oleh Telinga Manusia yaitu Frekuensi yang berkisar di antara 20Hz – 20.000Hz. Timbulnya suara dikarenakan adanya fluktuasi tekanan udara yang disebabkan oleh gerakan atau getaran suatu obyek tertentu. Ketika Obyek tersebut bergerak atau bergetar, Obyek tersebut akan mengirimkan Energi Kinetik untuk partikel udara disekitarnya. Hal ini dapat di-anologi-kan seperti terjadinya gelombang pada air. Sedangkan yang dimaksud dengan Frekuensi adalah jumlah getaran yang terjadi dalam kurun waktu satu detik. Frekuensi dipengaruhi oleh kecepatan getaran pada obyek yang menimbulkan suara, semakin cepat getarannya makin tinggi pula frekuensinya (Kho, 2014).



Gambar 2. 37 Prinsip Kerja *Speaker*

(Kho, 2014)

Pada gambar 2.37, dapat kita lihat bahwa pada dasarnya *Speaker* terdiri dari beberapa komponen utama yaitu *Cone, Suspension*, Magnet Permanen, *Voice Coil* dan juga Kerangka *Speaker*.

Dalam rangka menterjemahkan sinyal listrik menjadi suara yang dapat didengar, Speaker memiliki komponen Elektromagnetik yang terdiri dari Kumparan yang disebut dengan *Voice Coil* untuk membangkitkan medan magnet dan berinteraksi dengan Magnet Permanen sehingga menggerakan *Cone Speaker* maju dan mundur. *Voice Coil* adalah bagian yang bergerak sedangkan Magnet Permanen adalah bagian *Speaker* yang tetap pada posisinya. Sinyal listrik yang melewati *Voice Coil* akan menyebabkan arah medan magnet berubah secara cepat sehingga terjadi gerakan "tarik" dan "tolak" dengan Magnet Permanen. Dengan demikian, terjadilah getaran yang maju dan mundur pada *Cone Speaker*.

Cone adalah komponen utama *Speaker* yang bergerak. Pada prinsipnya, semakin besarnya *Cone* semakin besar pula permukaan yang dapat menggerakan udara sehingga suara yang dihasilkan *Speaker* juga akan semakin besar. *Suspension* yang terdapat dalam *Speaker* berfungsi untuk menarik *Cone* ke posisi semulanya setelah bergerak maju dan mundur. *Suspension* juga berfungsi sebagai pemegang *Cone* dan *Voice Coil.* Kekakuan (*rigidity*), komposisi dan desain *Suspension* sangat mempengaruhi kualitas suara *Speaker* itu sendiri (Kho, 2014).

2.2.18 OpenCV

2.2.18.1 Definisi OpenCV

OpenCV (Open Computer Vision) adalah program opensource berbasiskan C++ yang saat ini banyak digunakan sebagai program computer vision. Dengan OpenCV dapat membuat interaksi antara manusia dan computer, misalnya wajah dari manusia dideteksi oleh camera/webcam, lalu di proses oleh computer, untuk melakukan aksi tertentu seperti mengikuti/mengenal wajah orang tersebut. Kesemuanya itu membutuhkan openCV sebagai program utama antara webcam

dan perangkatnya yaitu computer maupun smartphone. Library ini terdiri dari fungsi-fungsi computer vision dan API (Aplication Programming Interface) untuk image processing high level maupun low level dan sebagai optimisasi aplikasi realtime. OpenCV sangat disarankan untuk programmer yang akan berkutat pada bidang computer vision, karena library ini mampu menciptakan aplikasi yang handal, kuat dibidang digital vision, dan mempunyai kemampuan yang mirip dengan cara pengolahan pada manusia (OpenCV, 2018).

2.2.18.2 Fitur OpenCV

Berikut ini adalah fitur-fitur pada library OpenCV :

• Manipulasi data gambar (alokasi memori, melepaskan memori, kopi gambar, setting serta konversi gambar)

• Image/video I/O (bisa menggunakan camera yang sudah didukung oleh library ini)

• Manipulasi matriks dan vector serta terdapat juga routines linear algebra (products, solvers, eigenvalues, SVD)

• *Image processing* dasar (filtering, *edge detection*, pendeteksian tepi, sampling dan interpolasi, konversi warna, operasi *morfologi, histograms, image pyramids*)

• Analisis structural

• Kalibrasi kamera

• Pendeteksian gerak

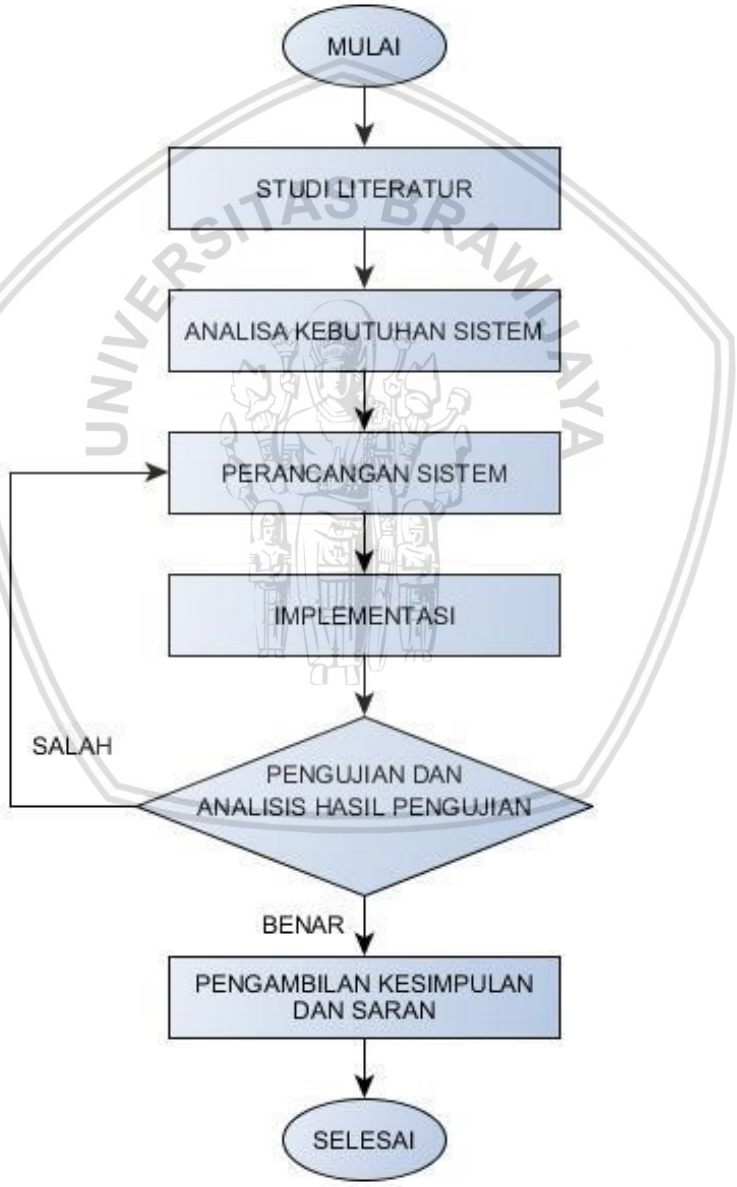
• Pengenalan objek

• Basic GUI (*Display* gambar/*video, mouse/keyboard control, scrollbar*)

• Image Labelling (*line, conic, polygon, test drawing*)

BAB III METODOLOGI PENELITIAN

Pada bab ini akan membahas mengenai langkah-langkah yang akan dilakukan dalam pembuatan sistem. Dimana terdapat tujuh langkah atau tahapan dalam pembuatan sistem ini diantaranya ialah: Analisa kebutuhan, perancangan, implementasi, pengujian aplikasi dan yang terakhir pengambilan kesimpulan serta saran sebagai referensi untuk pengembangan selanjutnya. Pada Gambar 3.1 akan dijelaskan langkah untuk melakukan penelitian secara umum.



Gambar 3. 1 Alur Penelitian

3.1 Studi Literatur

Studi Literatur merupakan penjelasan mengenai dasar teori yang dipakai sebagai pendukung dalam penelitian ini. Hal ini dapat membantu menambah pengetahuan penulis dalam mengerjakan penelitian ini. Teori pendukung didapat dari jurnal, paper, buku dan beberapa sumber internet. Dan beberapa teori pendukung penelitian ini ialah :

1. *Raspberry Pi*
2. *GPIO Raspberry Pi*
3. Motor DC
4. Citra Digital
5. Face Detection
6. *Face Recognition*
7. *Open CV*
   a. Macam-macam *library* yang tersedia pada *Open CV*
   b. Cara penggunaan *library*
   c. Cara pengimplementasian *library* pada Mikrokontroller

3.2 Analisis Kebutuhan Sistem

Analisis kebutuhan sistem pada penelitian ini ialah menganalisa semua kebutuhan terkait pembangunan sistem hingga sistem selesai dan siap dilakukan pengujian pada sistem. Dalam pembangunan sistem dibutuhkan analisis kebutuhan sistem meliputi kebutuhan perangkat lunak dan kebutuhan perangkat keras sistem.

3.2.1 Perangkat Keras

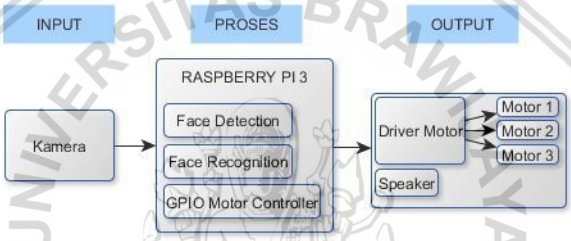
1. Laptop
2. *Raspberry Pi 3*
3. Motor DC
4. Kabel *Jumper*
5. *Project Board*
6. Kertas *PVC*
7. Roda
8. *Webcam/*kamera
9. Driver Motor L293D
10. *Blackbox*
11. Speaker
12. *Micro SD Memory*
13. Adapter

3.2.2 Perangkat Lunak

1. *Raspbian Jessie*
2. *Python 2.7*
3. *Open CV*
4. *Haarcascade Clasifier Library*
5. *Local Binary Pattern Histogram Library*
6. Numpy

3.3 Perancangan Sistem

Pada tahap perancangan, akan dibuat sebuah diagram yang menggambarkan sistem secara keseluruhan mencangkup perangkat keras maupun perangkat lunak. Berikut merupakan diagram sistem secara keseluruhan.

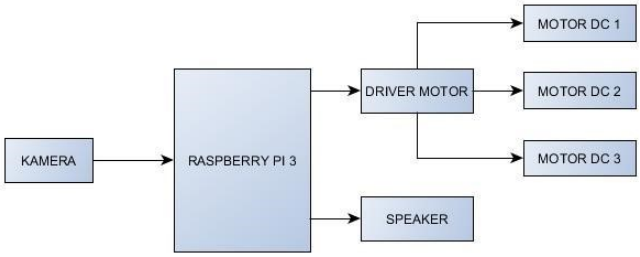


Gambar 3. 2 Diagram Block Sistem

Dari Gambar 3.2 perancangan sistem semuanya akan berpusat pada *raspberry pi* dimana *raspberry pi* nantinya akan menerima *input* berupa gambar dari *camera/webcam* yang akan di sambungkan via *USB*. *Face detection* dan *face recognition* disini akan ditanamkan pada *raspberry pi* dengan memanfaatkan kamera untuk menangkap gambar yang akan dijadikan *data set.* Dan ketika pengenalan wajah akan dilakukan pengambilan gambar dengan cara yang sama. Kemudian ada 3 yang akan menjadi keluaran dari sistem/*output* dari sistem yang pertama ialah *speaker* yang bertugas mengeluarkan suara dan menyebutkan siapakah yang terdeteksi di kamera, motor DC 1 yang bertugas melakukan *lock* dan *unlock* pintu dan motor DC 2 dan motor DC 3 yang bertugas melakukan membuka dan menutup pintu. Motor DC 2 dan 3 menjadi 1 proses *output* yang sama dikarenakan akan bertindak sebagai roda pintu yang bergerak dengan waktu dan arah yang sama. Sehingga untuk motor DC 2 dan 3 akan dihubungkan dalam satu output yang sama. Setalah melakukan 1 proses sistem akan kembali menjadi kondisi *stand by.*

3.3.1 Perancangan Perangkat Keras

Perancangan pada sisi perangkat keras melibatkan beberapa komponen antara lain ialah *webcam/camera, raspberry pi 3*, *IC L293D*, Motor DC*, speaker*, *adapter* dan *project board.* Berikut merupakan diagram perancangan pada sisi perangkat keras.

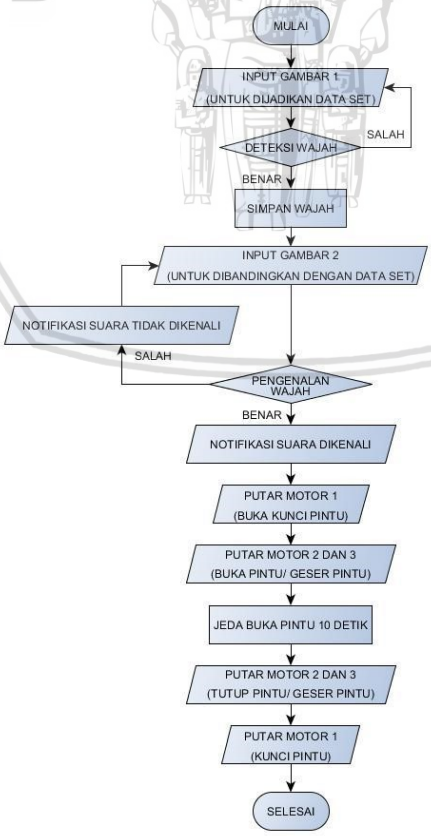


Gambar 3. 3 Diagram Perancangan Perangkat Keras

Pada Gambar 3.3 dapat dilihat bahwa kamera akan langsung dihubungkan melalui *usb port* pada raspberry pi 3 dan *power camera* juga mengambil dari raspberry pi melalui *adapter*. Proses keseluruhan akan dilakukan pada raspberry pi. Untuk output pergerakan motor tidak akan terhubung langsung dengan raspberry pi melainkan akan dihubungkan terlebih dahulu dengan driver motor model IC L293D . Namun untuk *output speaker* akan langsung dihubungkan dengan raspberry pi 3.

3.3.3 Perancangan Perangkat Lunak

Pada sistem perancangan perangkat lunak akan berfokus pada bagaimana sistem dapat mengenali wajah dan melakukan aksi setelah wajah terdeteksi dan dikenali. Berikut merupakan diagram alir sistem pada sisi perangkat lunak :



Gambar 3. 4 Diagram Perancangan Perangkat Lunak

Pada Gambar 3.4 Dapat dilihat bahwa pada sisi perangkat lunak difokuskan pada bagaimana sistem dapat mendeteksi wajah dan bagaimana sistem dapat mengenali wajah. Ketika wajah dikenali, sistem akan melakukan 3 keluaran aksi yang pertama ialah memberi notifikasi suara yang telah di simpan sebelumnya dan memutar motor DC 1 dan kemudian memutar motor DC 2 yang bersamaan dengan motor DC 3. Akan ada pemberian jeda 10 detik untuk seseorang masuk dan setalah masuk pintu akan kembali tertutup dan terkunci. Setelah itu sistem akan kembali menuju proses *input* wajah yang kedua yaitu *input* wajah untuk pengenalan.

3.4 Implementasi Sistem

Implementasi Sistem dilakukan sesuai dengan analisis kebutuhan dan perancangan sistem yang telah dibuat sebelumnya. Implementasi harus sesuai dengan gagasan awal sistem bagaimana sistem dapat bekerja. Pada tahap ini dilakukan secara berurutan mulai dari proses implementasi apakah yang mungkin untuk dimulai terlebih dahulu hingga implementasi yang mungkin diletakkan di akhir proses pembuatan sistem. Berikut merupakan urutan implementasi pembuatan sistem.

1. Implementasi *face detection* dan *face recognition* pada *personal computer*
   Pada penelitian dilakukan pembuatan *face detection* pada *personal computer* dengan menggunakan bahasa pemrograman yang sama dengan raspberry pi 3. Proses ini perlu dilakukan agar dalam mempermudah proses penanaman *face detection* dan *face recognition* pada *raspberry pi*.
2. Implementasi *face detection* dan *face recognition* pada *raspberry pi*
   Pada penelitian ini selanjutnya ialah menanamkan *face detection* dan *face recognition* pada raspberry pi 3 menggunakan *code* yang telah dibuat sebelumnya pada *personal computer.* Pada proses ini penulis tidak perlu memikirkan bagaimana mengenali wajah tetapi hanya fokus bagaimana cara menggunakan *library* yang sebelumnya telah digunakan pada *personal computer*. Hal ini sangat perlu dilakukan karena *personal computer* dan raspberry pi 3 memiliki arsitektur yang berbeda.
3. Implementasi desain skemantik *hardware*
   Pada penelitian ini selanjutnya ialah melakukan pengimplementasian desain skemantik *hardware* yang sudah dirancang sebelumnya. Dan menghubungkan semua komponen hardware menjadi satu bagian dan siap untuk dibuatkan prototipe. Komponen *hardware* akan dihubungkan dalam sebuah *project board*.
4. Menghubungkan *software* dengan *hardware*
   Pada penelitian ini selanjutnya ialah menghubungkan antara *software* dengan *hardware* yang akan digunakan seperti motor DC dan *Speaker.* Hal ini perlu dilakukan karena motor DC dan *Speaker* merupakan bagian dari keluaran sistem setelah wajah dapat dikenali. Untuk speaker akan dihubungkan melalui *port jack audio* yang tersedia dalam raspberry pi 3 sedangkan untuk motor DC akan dihubungkan melalui port *GPIO.*

5. Pembuatan Prototipe Pintu
   Pembuatan Prototipe sistem yaitu mengimplementasikan seluruh sistem pada sebuah desain prototipe pintu yang mana dapat membuat sistem lebih mudah untuk dimengerti dan dipresentasikan.

3.5 Analisis Pengujian Sistem

Pengujian pada sistem bertujuan untuk mengetahui apakah sistem sudah berjalan sesuai keinginan. Beberapa bagian sistem akan diuji sesuai dengan prosedur yang telah dibuat penulis. Beberapa simulasi pengujian yang akan dilakukan pada penelitian ini ialah.

1. Pengujian akurasi *face recognition* dengan menggunakan perbandingan 1 berbanding 4 antara penghuni rumah dan bukan penghuni rumah.
2. Pengujian pergerakan motor DC dengan simulasi pergerakan membuka dan menutup pintu serta *lock* dan *unlock.*
3. Pengujian keseluruhan sistem mulai dari pengujian perangkat keras maupun pengujian perangkat lunak.

3.6 Pengambilan Kesimpulan dan Saran

Pengambilan kesimpulan dilakukan setelah semua tahapan selesai dilakukan. Kesimpulan diambil berdasakan pada hasil pengujian dan analisis terhadap prototipe sistem yang telah diuji secara keseluruhan. Bagian terakhir dari penelitian ini ialah saran-saran yang nantinya dapat berguna untuk pengembangan aplikasi atau bisa juga menjadi referensi dalam pembuatan aplikasi serupa. Selain itu, saran juga digunakan untuk mempertimbangkan pengembangan aplikasi menjadi lebih baik.

BAB IV REKAYASA DAN KEBUTUHAN SISTEM

Pada bab ini bertujuan untuk menjelaskan secara rinci tentang rekayasa kebutuhan yang harus dipenuhi untuk perancangan hingga implementasi sistem. Sehingga diharapkan Implementasi *Face Recognition* Pada Raspberry Pi Untuk Sistem Keamanan *Smarthome* Dengan Pengolahan Citra Digital dapat berjalan sesuai dengan kebutuhan dan gagasan awal.

4.1 Gambaran Umum Sistem

Implementasi sistem otomatisasi pintu dengan *face recognition* menggunakan metode *haar-cascade* pada raspberry pi akan memiliki input berupa gambar atau citra yang didapat dari *webcam* dan di kombinasikan dengan *face detection*. Sistem akan menyimpan hasil *capture* yang pertama dalam sebuah *data set* dan disimpan dalam sebuah folder *data set*. Selanjutnya *data set* akan dilakukan proses *training* menggunakan LBPH *Recognition* yang kemudian akan menghasilkan keluaran berupa file yml. Selanjutnya *camera* akan berada pada posisi *live cam* dan mendeteksi setiap orang yang ada didepan kamera pada jarak tertentu. Jika kamera mengenali wajah yang terdeteksi sistem dengan cara membandingkan dengan wajah yang telah disimpan sebelumnya maka sistem akan menyebutkan siapa yang terdeteksi melalui media *speaker,* kemudian membuka kunci dan membuka pintu dengan selang waktu yang telah ditentukan. Sistem akan memberikan waktu 10 detik bagi yang wajahnya dikenali untuk memasuki rumah. Jika melebihi 10 detik sistem akan menutup kembali pintu, mengunci pintu dan pengguna harus mengulangi proses pengenalan wajah jika ingin memasuki rumah.

4.1.1 Perspektif Sistem

Ada beberapa poin yang harus dicapai sistem agar sistem dapat dikatakan berhasil. Sistem harus bisa mendeteksi keberadaan muka ketika ada seseorang yang berada di depan kamera sistem. Sistem juga harus bisa mengenali wajah sesuai dengan *data set* yang telah disimpan sebelumnya. Ketika sudah mengenali sistem juga harus menyebutkan siapa yang terdeteksi kemudian membukakan kunci dan membukakan pintu. Setelah mengenali, sistem harus memberikan tenggang waktu untuk masuk selama 10 detik dan menutup kembali pintu jika melebihi 10 detik. Sistem juga tidak boleh membukakan pintu apabila wajah yang terdeteksi tidak sesuai dengan *data set* yang tersimpan sebelumnya, karena itu akan masuk dalam kategori eror sistem.

4.1.2 Ruang Lingkup

Ruang lingkup sistem ini ialah sistem diimplementasikan pada mikrokontroller raspberry pi 3. Untuk kamera, sistem akan menggunakan *Webcam* dan tidak menggunakan *pi camera*. Sistem menggunakan 3 buah motor DC dengan *power* menggunakan adapter sebesar 12 volt. Sistem berfokus pada akurasi pengenalan

wajah dan juga berfokus pada bagaimana menghubungkan *face recognition* dengan sistem buka tutup pintu dan sistem penguncian pintu serta informasi berupa suara yang menyebutkan siapa yang terdeteksi.

4.1.3 Karakteristik Pengguna

Sistem ini diperuntukan untuk rumah berpenghuni yang belum memiliki sistem keamanan rumah atau ingin menambahkan sistem keamanaan rumah. Harapannya sistem ini dapat membantu mengamankan rumah dari tindakan pencurian. Sistem juga bisa diperuntukan untuk sistem absensi kelas namun perlu dilakukan pengembangan pada saat pembuatan *data set* agar dapat menampung banyak *data set* dan dilakukan sedikit perubahan perancangan pada *output.*

4.1.4 Lingkungan Operasi Sistem

Ada beberapa persyaratan lingkungan yang dapat mendukung kebutuhan ataupun kinerja dari sistem :

1. Pintu yang digunakan merupakan pintu yang digeser, hal ini dapat mempermudah dalam peletakan motor DC yang bertugas untuk membuka dan menutup pintu.
2. Kondisi pencahayaan harus cukup, Semakin jelas gambar yang tertangkap oleh *camera* dapat mempengaruhi kecepatan sistem dalam melakukan pengenalan wajah.
3. Sistem sebaiknya digunakan pada rumah yang tidak tepat berada dipinggir dijalan raya (opsional), hal ini dapat memaksimalkan sistem dalam penyebutan siapa yang dikenali. Kondisi bising dikhawatirkan dapat membuat suara jadi tidak jelas terdengar.

4.1.5 Batasan Perancangan Dan Implementasi Sistem

Penelitian ini membutuhkan batasan pada saat perancangan dan implementasi sistem agar penelitian menjadi lebih terarah dan tidak keluar dari konteks yang telah dijelaskan. Berikut merupakan batasan sistem.

1. Sistem nantinya akan menjadi sebuah sistem keamanan yang berfokus pada sistem penguncian dan buka tutup pintu rumah serta pengenalan wajah.
2. Sistem akan menggunakan 3 buah motor yang memiliki fungsi berbeda, motor 1 digunakan untuk menggerakan kunci dan motor 2 serta 3 digunakan untuk menggerakan pintu.
3. Speaker yang digunakan untuk menyampaikan informasi sistem ialah speaker biasa yang tidak membutuhkan *driver* dalam penggunaannya dan *Audio* yang digunakan ialah *audio* yang siap pakai sehingga penelitian ini tidak menjelaskan proses *audio*.
4. Pengidentifikasian pengguna sistem menggunakan *Haar Cascade Clasifier* sebagai pendeteksi dan menggunakan *Local Binary Pattern Histogram* sebagai pengenalan yang ditanamkan pada raspberry pi 3.

Perangkat yang digunakan :

1. Kamera menggunakan *WebCamera* dengan resolusi 2 *megapixel*
2. Mikrokontroller menggunakan Raspberry pi 3
3. Motor DC 6V
4. Driver Motor menggunakan L293D
5. Speaker
6. Adapter 12V

Output :

Output jika wajah dikenali ialah sistem akan menyebutkan siapa yang terdeteksi kemudian membuka kunci dan membukakan pintu dengan cara memutar motor searah dengan arah jarum jam dan memutar berlawanan jika sudah melebihi waktu tunggu yaitu 10 detik untuk mengunci dan menutup pintu kembali. *Output* jika wajah tidak dikenali ialah sistem akan menyebutkan bahwa wajah tidak dikenali dan kembali pada fase *Live cam* tanpa membuka kunci dan pintu.

4.1.6 Asumsi dan Ketergantungan

Beberapa asumsi dan ketergantungan dalam penerapan sistem ini adalah sebagai berikut :

1. Diasumsikan posisi seseorang yang akan dideteksi berada pada posisi yang telah ditentukan.
2. Diasumsikan *driver webcam* sudah terinstall terlebih dahulu pada mikrokontroller.
3. Keadaan awal pintu berada pada posisi tertutup.
4. Setiap orang akan melakukan pendeteksian harus secara bergantian satu persatu didepan camera sistem.
5. Audio yang digunakan sebagai keluaran suara pada *speaker* dianggap sudah siap pakai.

4.2 Rekayasa Kebutuhan

Pada sub bab ini menjelaskan seluruh kebutuhan agar sistem dapat bekerja sesuai dengan tujuan penelitian ini. Pada sub bab ini menjelaskan kebutuhan antarmuka sistem yang meliputi antarmuka Pengguna, antar muka perangkat lunak dan antar muka perangkat keras. Kebutuhan fungsional dan non-fungsional sistem dan kebutuhan lainnya. Berikut merupakan penjelasan dari rekayasa kebutuhan sistem :

4.2.1 Kebutuhan Antarmuka

4.2.1.1 Antarmuka Pengguna

Antarmuka untuk sisi pengguna tidak didesain menggunakan *GUI* karena pengguna hanya tinggal mengklik 2 kali pada masing-masing *code* ketika ingin melakukan pembuatan *data set , training data set*, dan memulai sistem.

4.2.1.2 Antarmuka Perangkat Keras

Perangkat keras didesain agar terlihat menarik dengan membuat prototipe pintu menggunakan kertas PVC. Berikut beberapa hal yang dibutuhkan dalam pembuatan antarmuka perangkat keras :

1. Raspberry pi 3
2. Webcam
3. Driver Motor L293D
4. Motor DC
5. Speaker
6. Kertas PVC
7. Kabel Jumper
8. Blackbox
9. Heatsink
10. Adapter 12V
11. Lem
12. Project Board

Sistem akan menggunakan pintu geser dalam implementasi. Hal ini dilakukan untuk mempermudah dalam peletakan motor pada pintu dimana motor akan diletakan sebagai roda pada pintu. Sistem akan menggunakan motor DC dikarenakan sistem tidak membutuhkan perhitungan RPM(*Rotation Per Minute*) Dan juga sistem tidak melakukan percepatan ketika menggerakan pintu. Penggunaan motor DC merupakan cara paling efektif dibandingkan dengan servo dikarenakan fungsi motor disini hanya sebatas membuka dan menutup pintu dengan jangka waktu yang telah ditentukan.

4.2.1.3 Antarmuka Perangkat Lunak

Kebutuhan antarmuka perangkat lunak sistem terdiri dari beberapa perangkat yaitu :

1. Raspbian Jessie
2. Python 2.7
3. Numpy
4. PIP
5. Open CV
6. *Haarcascade Library*
7. *LBPH Library*
8. *Webcam Driver*
9. *Audio format WAV*

Sistem akan menggunakan *library* open cv untuk proses pendeteksian dan proses pengenalan. Open cv merupakan *library* untuk mempermudah seseorang dalam mengembangkan kemampuannya di dalam disiplin ilmu *computer vision*. Open cv yang digunakan oleh sistem tersedia gratis dan juga dapan di *download* pada *website official* open cv. Penggunaan *library* tersebut merupakan cara yang paling efektif mengingat perangkat yang digunakan ialah sebuah mikrokontroller.

4.2.2 Kebutuhan Fungsional

Pada kebutuhan fungsional sistem menjelaskan mengenai bagaimana sistem memberikan layanan kepada pengguna sesuai perancangan yang sudah dibuat sebelumnya. Kebutuhan fungsional pada sistem ini adalah sebagai berikut.

1. Sistem dapat melakukan pengambilan gambar dengan menggunakan webcam.
   a. Penjelasan
      Ini merupakan fungsi dasar yang sangat wajib terpenuhi yaitu *system* harus dapat melakukan pengambilan gambar melalui *webcam* yang telah terinstall pada *raspberry pi*. Dan pengambilan gambar dapat dikontrol melalui *raspberry pi* penggunakan perintah pada Bahasa pemrograman *python* 2.7.
   b. Stimulus
      Ketika webcam terhubung pada *raspberry pi* buka *python idle* untuk melakukan pengambilan gambar menggunakan beberapa *syntax* pada *python*. Gambar yang telah diambil dapat kita lihat pada folder yang telah di tentukan sebelumnya. Dan hasilnya dapat dibuka dan di lihat menggunakan *preview* foto pada *raspberry pi.*
   c. Kebutuhan Fungsional
      Fungsi ini wajib terpenuhi karena merupakan fungsi dasar dan wajib sistem yaitu melakukan pengambilan gambar menggunakan *webcam* yang sudah *terinstall* sebelumnya.
2. Sistem dapat melakukan pembuatan *data set* foto.
   d. Penjelasan
      Sebelum sistem siap melakukan pengenalan wajah. Sistem harus bisa melakukan pendeteksian wajah terlebih dahulu dan disimpan dalam sebuah *data set*. Pembuatan *data set* dilakukan dengan menggunakan webcam yang telah tersambung dengan *raspberry pi*. Sistem hanya akan menyimpan bagian wajahnya saja, bukan keseluruhan hasil *capture camera*. Kemudian semua data wajah disimpan pada sebuah folder dan memberikan kode unik pada setiap wajah untuk membedakan antara wajah yang satu dengan yang lainnya.
   e. Stimulus
      Ketika semua sistem sudah dihubungkan satu sama lain dan telah tersambung dengan catu daya. Masuk ke dalam folder dan jalankan *code create data set*. Sebelum menjalankan *code,* bersiaplah didepan camera karena sistem akan dengan otomatis menghidupkan *webcam* dan mendeteksi wajah dengan bantuan *library open CV*. Hasil *capture* dapat dilihat pada folder *data set.*
   f. Kebutuhan Fungsional
      Fungsi ini wajib terpenuhi karena *face recognition* merupakan sistem yang membandingkan wajah dengan wajah yang telah ter-*capture* dan tersimpan sebelumnya.

3. Sistem dapat melakukan *training data set* foto wajah yang telah tersimpan sebelumnya.
   a. Penjelasan
      Sistem dapat mengubah foto yang telah tersimpan pada folder data set ke dalam matrix dan menyimpan hasil perhitungan matrix pada sebuah file dengan format yml.
   b. Stimulus
      Jalankan *code training* gambar maka sistem akan dengan cepat mengindentifikasi gambar dan mengkonversinya ke dalam matrix dan meletakkannya dalam sebuah folder.
   c. Kebutuhan Fungsional
      Fungsi ini wajib terpenuhi karena hasil dari *training* data ini yang akan dibandingkan dengan gambar wajah yang terdeteksi saat *live camera.*
4. Sistem dapat melakukan *face recognition*
   a. Penjelasan
      Sistem dapat mengenali wajah siapa saja yang ada pada camera saat *live cam*. Sistem akan melakukan pengulangan secara terus menerus dan melaporkan langsung apakah wajah dikenali ataupun tidak dikenali.
   b. Stimulus
      Ketika kode dijalankan sistem akan langsung melakukan pengecekan wajah dan melakukan perulangan perbandingan dengan data wajah yang telah tersimpan sebelumnya.
   c. Kebutuhan Fungsional
      Fungsi ini juga sangat penting dikarenakan inti dari sistem ialah *face recognition*. Sistem dapat mengenali apakah dia penghuni ataukah bukan.
5. Sistem dapat memainkan *audio* dan memutar motor
   a. Penjelasan
      Setelah mengenali wajah sistem akan melakukan aksi berupa pemutaran *audio* siapa yang terdeteksi dan suara akan dikeluarkan melalui *speaker*. Kemudian membuka kunci dan membukakan pintu setelah itu menutupnya kembali ketika pengguna sudah di dalam.
   b. Stimulus
      Pengguna berdiri tegak didepan sistem menghadap *camera.* Jika pengguna dikenali sistem akan memutar motor 1 searah jarum jam kunci akan terbuka. Kemudian sistem akan memutar motor 2 dan 3 secara bersamaan searah jarum jam dan pintu akan bergeser terbuka otomatis. Pengguna akan diberi waktu 10 detik untuk masuk. Setelah itu motor akan berputar kembali melawan arah jarum jam untuk untuk menutup dan mengunci pintu.

c. Kebutuhan Fungsional
Fungsi ini sangatlah penting karena ini merupakan inti dari sistem dan harus berjalan sesuai dengan gagasan awal demi menjaga keamanan rumah pengguna.

4.2.3 Kebutuhan Non Fungsional

Pada tahap ini ada beberapa kebutuhan yang dapat memaksimalkan kinerja sistem. Antara lain ialah :

1. Kebutuhan *Power Consumption*
   Ada dua buah sumber daya yang digunakan untuk menjalankan sistem , yang pertama ialah sumber daya yang didapat melalui adapter untuk menjalankan raspberry pi 3 dan yang kedua ialah sumber daya yang didapat melalui adapter 12 volt.
   a. Power Consumption Raspberry Pi 3
   Standar *power* untuk *raspberry pi* 3 ialah 5V 2,5A hal ini dapat dilihat pada datasheet *raspberry pi.* Jika power berada dibawah itu maka raspberry tidak akan berjalan secara maksimal dengan ditandai munculnya gambar peringatan pada pojok kanan atas *Raspbian jessie*. Agar sistem dapat berjalan maksimal maka adapter yang digunakan untuk sumber daya *raspberry pi* 3 ialah 5V 3A. Hal ini dapat dilihat pada *datasheet raspberry pi* 3.
   b. *Power Consumption* Motor DC
   Untuk masukan tegangan motor DC standarnya ialah 12V, jika tegangan untuk motor DC kurang dari itu maka yang akan terjadi ialah motor tidak akan bekerja secara maksimal dan dikhawatirkan motor tidak kuat untuk membuka dan menutup pintu secara otomatis. Untuk memaksimalkan perputaran motor DC maka diperlukan adapter 12v yang nantinya akan dimasukan pada IC L293D. Pada datasheet IC, IC akan membagi 2 tegangan yang masuk untuk memenuhi tegangan masing-masing motor. Dengan menggunakan adapter 12v akan memaksimalkan putaran motor dan kelebihan tegangan yang diberikan masih dalam batas wajar menurut informasi *data set* IC L293D.

BAB V PERANCANGAN DAN IMPLEMENTASI

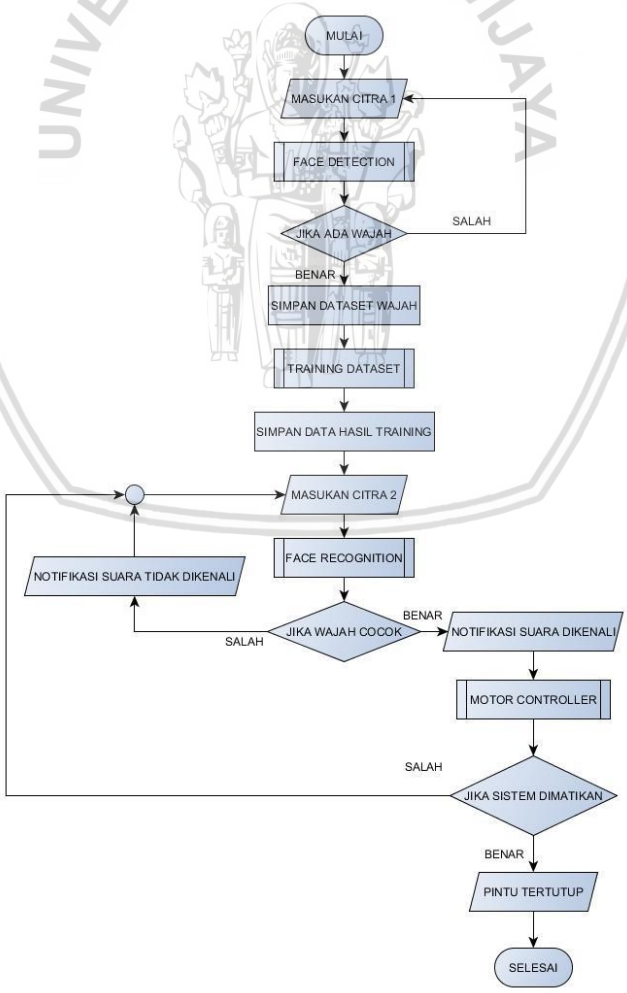
Pada bab ini akan dijelaskan bagaimana perancangan dan implementasi sistem secara terperinci mulai dari perancangan pada sisi perangkat lunak maupun pada sisi perangkat keras.

5.1 Perancangan Sistem

Tahap perancangan sistem dibagi menjadi 2 tahapan yang pertama ialah perancangan pada sisi perangkat lunak dan yang selanjutnya ialah perancangan pada sisi perangkat keras.

5.1.1 Perancangan Perangkat Lunak

Perancangan perangkat lunak dilakukan dalam beberapa tahap, dan setiap tahap akan dijelaskan secara terperinci pada masing-masing sub bab. Perancangan perangkat lunak akan dijelaskan menggunakan *flowchart* program. Gambar 5.1 merupakan diagram alir perancangan perangkat lunak secara keseluruhan.



Gambar 5. 1 Diagram Alir Perancangan Perangkat Lunak

5.1.1.1 *Face Detection*

Perancangan *Face Detection* atau sering dikenal dengan pendeteksian wajah pada penelitian ini menggunakan metode *Haar Cascade Classifier*. Metode *Haar Cascade* digunakan karena merupakan metode yang sangat ringan dan memiliki kecepatan pendeteksian yang cepat. Penulis menggunakan Library yang disediakan oleh Open CV yaitu *haarcascadefrontal_face_default.xml*. Library tersebut sudah mencangkup semua proses deteksi wajah dan berikut merupakan urutan cara kerja library Open CV yaitu *haarcascadefrontal_face_default.xml.*

Pertama-tama citra harus dikonversi dari RGB menjadi *Grayscale.* Secara teori cara mengkonversikannya ialah dengan menggunakan rumus sebagai berikut.

$$Grayscale = 0.2989\ \mathrm{R} + 0.5870\ \mathrm{G} + 0.1140\ \mathrm{B} \quad \textbf{(5.1)}$$

Pada persamaan 5.1 kita dapat melihat bahwa setiap nilai R nilai G dan nilai B pada setiap piksel akan dikalikan dengan nilai sesuai rumus agar sebuah citra dapat berubah menjadi *grayscale.* Sebagai contoh, gambar 5.2 merupakan citra RGB dengan dimensi 200 x 150.

Gambar 5. 2 Citra RGB

Kemudian penulis akan mengakses sebuah piksel secara *random* untuk mengetahui nilai RGB salah satu piksel. Pada piksel *x* = 100 dan *y* = 100 diketahui Nilai RGB = 57 34 16. Dari data tersebut maka dapat dihitung *Grayscale* dengan cara sebagai berikut :

$$\begin{aligned}
\text{Diketahui} = \quad & \textit{Red}\,(\text{merah}) = 57 \\
& \textit{Green}\,(\text{hijau}) = 34 \\
& \textit{Blue}\,(\text{biru}) = 16 \\
Grayscale &= 0.2989\ \mathrm{R} + 0.5870\ \mathrm{G} + 0.1140\ \mathrm{B} \\
Grayscale &= (0.2989 \text{ x } 57) + (0.5870 \text{ x } 34) + (01140 \text{ x } 16) \\
Grayscale &= 17.0373 + 19.958 + 1.824 \\
Grayscale &= 38.8193
\end{aligned}$$

Citra *Grayscale* merupakan citra dengan ukuran 8 bit. Sehingga memiliki $(2^8 - 1)$ kemungkinan warna mulai dari 0 hingga 255 dimana 0 berarti hitam dan 255 berarti putih. Warna abu memiliki *range* dari 1 hingga 254 dimulai dari abu paling gelap hingga abu terang dan akhirnya mendekati putih. Jadi hasil *grayscale* = 38.8193 merupakan warna abu-abu yang cenderung mendekati hitam. Kemudian aplikasikan rumus tersebut di semua piksel dan hasil dari konversi citra RGB pada gambar 5.2 akan menjadi seperti pada gambar 5.3 berikut :

Gambar 5. 3 Citra *Grayscale*

Dari gambar 5.3 dapat dilihat bahwa setelah dikonversi menjadi citra *Grayscale* sebagian besar daerah pada citra memiliki nilai yang cenderung hitam. Proses selanjutnya ialah menentukan *Haar feature* yaitu metode yang membutuhkan training terlebih dahulu untuk mendapatkan suatu pohon keputusan apakah di *frame* tersebut ada objek atau tidak. Ada banyak fitur haar yang dapat diimplementasikan, untuk lebih lengkapnya dapat dilihat pada bab 2. *Haar feature* digunakan untuk mencari posisi wajah dengan cara mencari fitur-fitur dengan tingkat pembeda yang tinggi. Selisih dari nilai fitur yang diimplementasikan akan dijadikan *threshold* klasifikasi ada objek atau tidak. Yang dalam penelitian ini objek yang dicari adalah wajah. Berikut cara meletakan Fitur *Haar* dalam sebuah citra.

Gambar 5. 4 *Haar Feature*

Pada gambar 5.4 penulis meletakan sebuah *feature haar* pada gambar hasil konversi *grayscale*. *Feature* yang digunakan berupa *line feature* yang merupakan satu dari banyak *haar feature*. Untuk menentukan nilai *feature* tersebut digunakan persamaan berikut :

$$NILAI\ FITUR = |(total\ piksel\ hitam) - (total\ piksel\ putih)| \qquad \textbf{(5.2)}$$

Pada persamaan 5.2 dibutuhkan nilai piksel, ada banyak cara untuk mengetahui nilai piksel . Namun dalam kasus ini dibutuhkan Teknik yang dapat dilakukan dengan sangat cepat yang dapat diimplementasikan pada ratusan fitur *haar* dengan skala yang berbeda-beda. Salah satu teknik yang cukup efisien untuk melakukan itu ialah *Integral Image. Integral Image* ialah sebuah citra yang tiap piksel nya merupakan akumulasi dari piksel atas dan kirinya. Berikut merupakan rumus *Integral Image* :

$$S(x,y) = i\,(x,y) + S((x-1),y) + S(\,x,(\,y-1)) - S\,((x-1),(y-1)) \quad \textbf{(5.3)}$$

Pada persamaan 5.3 *i(x,y)* merupakan nilai asli matrik citra. Gambar 5.4 menunjukan bahwa *feature* diletakan pada tengah wajah yang memiliki nilai matrix sebagai berikut :

$$\begin{matrix} 2 & 1 & 2 & 7 & 4 & 8 \\ 4 & 8 & 2 & 4 & 1 & 7 \\ 3 & 4 & 6 & 5 & 2 & 7 \\ 4 & 9 & 5 & 8 & 8 & 6 \\ 5 & 8 & 2 & 5 & 6 & 8 \\ 4 & 5 & 1 & 6 & 2 & 7 \end{matrix}$$

Matrik tersebut merupakan nilai *grayscale* dari setiap *pixel* pada citra. Kemudian fitur *haar* yang diimplementasikan ialah seperti berikut :

$$\begin{matrix} 2 & 1 & 2 & 7 & 4 & 8 \\ 4 & 8 & 2 & 4 & 1 & 7 \\ 3 & 4 & 6 & 5 & 2 & 7 \\ 4 & 9 & 5 & 8 & 8 & 6 \\ 5 & 8 & 2 & 5 & 6 & 8 \\ 4 & 5 & 1 & 6 & 2 & 7 \end{matrix}$$

Untuk mengetahui nilai fitur *haar* rumus yang digunakan ialah rumus pada persamaan 5.2. Oleh sebab itu pertama-tama harus diketahui terlebih dahulu nilai masing-masing bagian fitur. Jika menggunakan penjumlahan manual untuk menghitung nilai piksel masing-masing fitur ialah dengan menjumlahkan semuanya seperti pada contoh berikut ini :

$$Daerah\ Hitam\ 1 = \begin{matrix} 2 & 1 & 2 & 7 & 4 & 8 \\ 4 & 8 & 2 & 4 & 1 & 7 \end{matrix}$$

$$Daerah\ Putih\ 1 = \begin{matrix} 3 & 4 & 6 & 5 & 2 & 7 \\ 4 & 9 & 5 & 8 & 8 & 6 \end{matrix}$$

$$Daerah\ Hitam\ 2 = \begin{matrix} 5 & 8 & 2 & 5 & 6 & 8 \\ 4 & 5 & 1 & 6 & 2 & 7 \end{matrix}$$

$$Daerah\ Hitam\ 1 = 2 + 1 + 2 + 7 + 4 + 8 + 4 + 8 + 2 + 4 + 1 + 7$$

$$Daerah\ Hitam\ 1 = 50$$

$$Daerah\ Hitam\ 2 = 5 + 8 + 2 + 5 + 6 + 8 + 4 + 5 + 1 + 6 + 2 + 7$$

$$Daerah\ Hitam\ 2 = 59$$

$$Daerah\ Putih\ 1 = 3 + 4 + 6 + 5 + 2 + 7 + 4 + 9 + 5 + 8 + 8 + 6$$

$$Daerah\ Putih\ 1 = 67$$

$$NILAI\ FITUR = |(total\ piksel\ hitam) - (total\ piksel\ putih)|$$

$$NILAI\ FITUR = |(50 + 59) - (67)|$$

$$NILAI\ FITUR = 42$$

Cara perhitungan diatas merupakan cara perhitungan manual untuk mencari nilai fitur. Namun cara tersebut tidak akan efektif dan efisien jika ada ratusan jenis fitur *haar* dan juga ratusan jenis ukuran serta posisi fitur yang *random* dalam citra. Jika menghitung dengan menggunakan *integral image* akan lebih efisien dan dapat sekali jalan. Berikut merupakan perhitungan menggunakan *Integral Image* :

1. Siapkan matrik dengan ukuran yang sama yang akan bertindak sebagai *buffer*. Fungsi dari matrik *buffer* ialah sebagai tempat untuk meletakan nilai *integral image* dari tiap piksel.

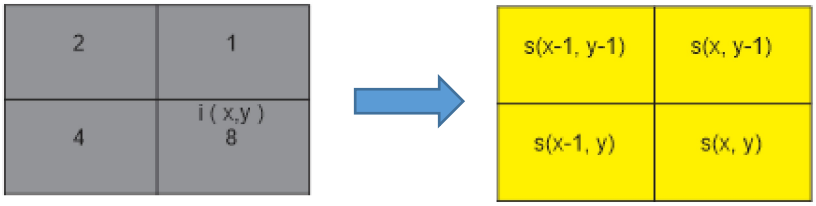
0	0	0	0	0	0
0	0	0	0	0	0
0	0	0	0	0	0
0	0	0	0	0	0
0	0	0	0	0	0
0	0	0	0	0	0

2. Buatlah *summed area table* sebagai contoh seperti pada gambar 5.5 berikut. Fungsi *summed area table* ialah untuk mempermudah proses perhitungan.

Gambar 5. 5 ilustrasi area *s (x,y)*

S (x,y) merupakan nilai *summed area* pada bidang *(x,y),* nilai *i(x,y)* merupakan intensitas dari citra asli, *s (x, y-1)* merupakan nilai *summed area* dari nilai piksel tetangga atas atau tetangga *(y-1).* Dan *s(x-1,y)* merupakan nilai *summed area* dari nilai piksel tetangga *x* serta *s(x-1, y-1)* merupakan nilai *summed area*

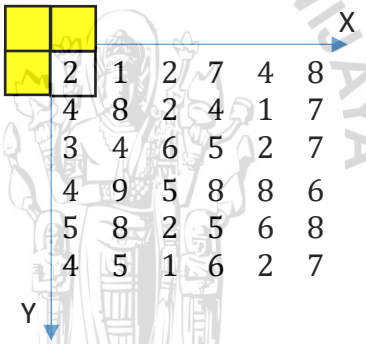
dari nilai piksel tetangga diagonalnya. Jika diilustrasikan dalam gambar akan menjadi seperti gambar 5.6.



Gambar 5. 6 Summed Area Table

Gambar 5.6 menunjukan bahwa matrik dengan *background* abu merupakan matrik asli dari citra dan *background* kuning merupakan nilai *integral image* yang akan dicari.

3. Selanjutnya mencari nilai *integral image* dengan rumus pada persamaan 5.3 pada tiap piksel pada citra yang mana ialah tiap nilai pada matrik asli yang telah diketahui dan meletakkan nilainya pada matrik *buffer* yang telah disediakan.



$$S(x,y) = i\,(x,y) + S((x-1),y) + S(\,x,(\,y-1)) - S\,((x-1),(y-1))$$
$$S(1,1) = 2 + 0 + 0 - 0$$
$$S(1,1) = 2$$

Kemudian lakukan perhitungan pada semua piksel. Dan setiap hasil perhitungan diletakan pada *buffer*.

$$S(x,y) = i\,(x,y) + S((x-1),y) + S(\,x,(\,y-1)) - S\,((x-1),(y-1))$$
$$S(2,1) = 1 + 2 + 0 - 0$$
$$S(2,1) = 3$$
$$S(x,y) = i\,(x,y) + S((x-1),y) + S(\,x,(\,y-1)) - S\,((x-1),(y-1))$$
$$S(1,2) = 4 + 2 + 0 - 0$$
$$S(1,2) = 6$$

$$S(x,y) = i\,(x,y) + S((x-1),y) + S(\,x,(\,y-1)) - S\,((x-1),(y-1))$$
$$S(2,2) = 1 + 2 + 0 - 0$$
$$S(2,2) = 15$$

Setelah diletakan setiap nilai pada matrik *buffer*, maka *buffer* akan menjadi seperti matrik berikut :

$$\begin{matrix} 2 & 3 & 0 & 0 & 0 & 0 \\ 6 & 15 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{matrix}$$

Dan setelah semua piksel telah dihitung maka matrik *integral image* akan menjadi seperti berikut :

$$\begin{matrix} 2 & 3 & 5 & 12 & 16 & 24 \\ 6 & 15 & 19 & 30 & 35 & 50 \\ 9 & 22 & 32 & 48 & 55 & 77 \\ 13 & 35 & 50 & 74 & 89 & 117 \\ 18 & 48 & 65 & 94 & 115 & 151 \\ 22 & 57 & 75 & 110 & 133 & 176 \end{matrix}$$

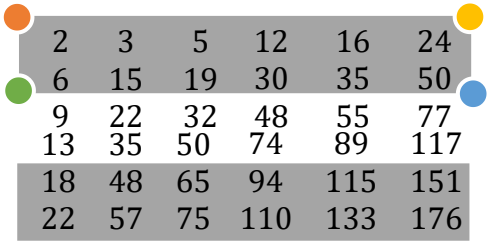
Setelah ditemukan nilai *integral image*, selanjutnya bagi daerah *integral* matrix sesuai dengan fitur *haar* sebelumnya.

$$\begin{matrix} 2 & 3 & 5 & 12 & 16 & 24 \\ 6 & 15 & 19 & 30 & 35 & 50 \\ 9 & 22 & 32 & 48 & 55 & 77 \\ 13 & 35 & 50 & 74 & 89 & 117 \\ 18 & 48 & 65 & 94 & 115 & 151 \\ 22 & 57 & 75 & 110 & 133 & 176 \end{matrix}$$

Untuk menentukan jumlah nilai piksel pada suatu area ialah menggunakan persamaan 5.4.

$$Luas\ area = L_1 + L_4 - (L_2 + L_3) \qquad \textbf{(5.4)}$$

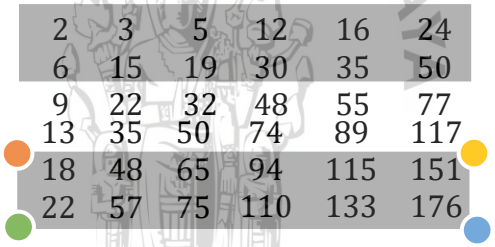
a. Luas daerah hitam 1



2	3	5	12	16	24
6	15	19	30	35	50
9	22	32	48	55	77
13	35	50	74	89	117
18	48	65	94	115	151
22	57	75	110	133	176

Diketahui = $L_1 = Orange = 0$
$L_2 = Kuning = 0$
$L_3 = Hijau = 0$
$L_4 = Biru = 50$
Luas area $= L_1 + L_4 - (L_2 + L_3)$
Luas area $= 0 + 50 - (0 + 0)$
Luas area $= 50$

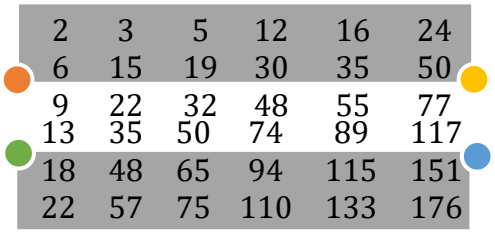
b. Luas daerah hitam 2



2	3	5	12	16	24
6	15	19	30	35	50
9	22	32	48	55	77
13	35	50	74	89	117
18	48	65	94	115	151
22	57	75	110	133	176

Diketahui = $L_1 = Orange = 0$
$L_2 = Kuning = 117$
$L_3 = Hijau = 0$
$L_4 = Biru = 176$
Luas area $= L_1 + L_4 - (L_2 + L_3)$
Luas area $= 0 + 176 - (117 + 0)$
Luas area $= 59$

c. Luas daerah putih 1



2	3	5	12	16	24
6	15	19	30	35	50
9	22	32	48	55	77
13	35	50	74	89	117
18	48	65	94	115	151
22	57	75	110	133	176

Diketahui = $L_1 = Orange = 0$
$L_2 = Kuning = 50$
$L_3 = Hijau = 0$
$L_4 = Biru = 117$
Luas area $= L_1 + L_4 - (L_2 + L_3)$
Luas area $= 0 + 117 - (50 + 0)$
Luas area $= 67$

$$NILAI\ FITUR = |(total\ piksel\ hitam) - (total\ piksel\ putih)|$$
$$NILAI\ FITUR = |(50 + 59) - (67)|$$
$$NILAI\ FITUR = 42$$

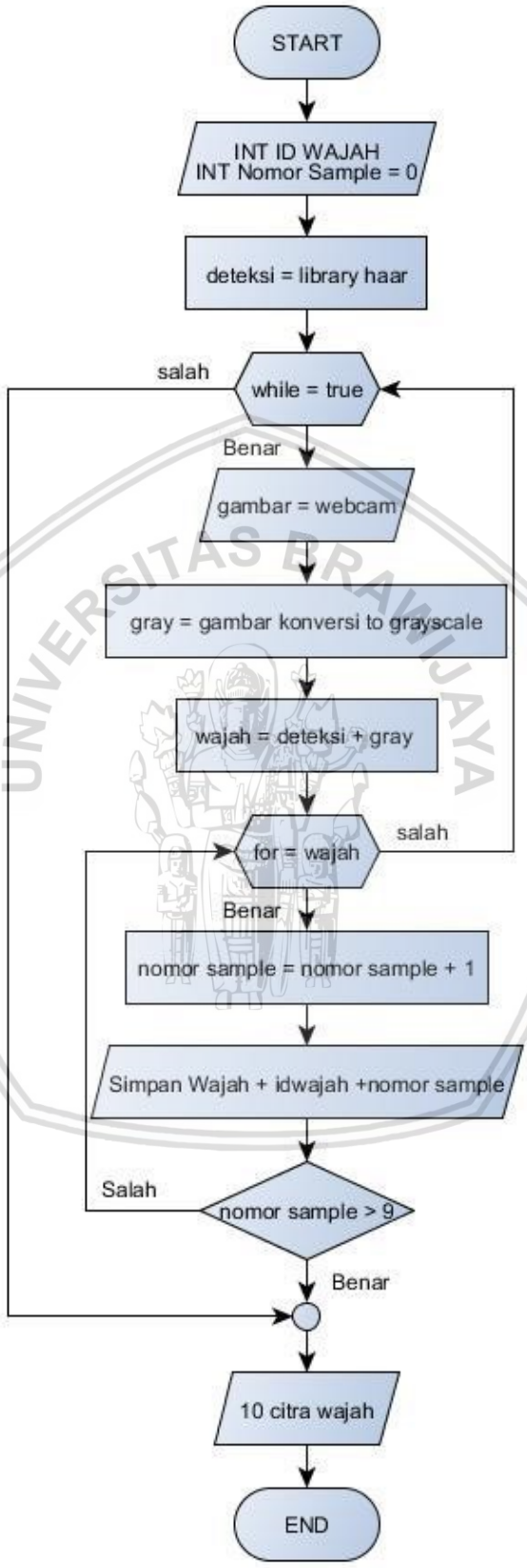
Perhitungan menggunakan *integral image* terbukti sama dengan perhitungan secara manual. Nilai fitur akan digunakan untuk menentukan ada tidaknya objek dan setiap objek yang akan dideteksi harus memiliki *threshold* atau batas ambang nilai fitur.

Setelah pencarian nilai fitur proses terakhir ialah membuat *cascade classifier*. Kita tahu Haar like features memiliki sifat *learner* dan *classifier* yang lemah. Jadi unrtuk menambah akurasi perlu dilakukan proses *haar-like feature* secara massal. Proses *haar like feature* yang dilakukan secara masal dan terorganisir disebut dengan *cascade classifier*. Biasanya dilakukan menggunakan *stage* filter dengan jumlah fitur yang berbeda-beda. Jika nilai fitur tidak memenuhi maka hasil akan langsung ditolak. Contoh sederhana *cascade classifier* ialah sebagai berikut :

1. *Stage* Filter 1 (3 fitur *haar*)
2. *Stage* Filter 2 (5 fitur *haar*)
3. *Stage* Filter 3 (10 fitur *haar*)
4. *Stage* Filter 4 (20 fitur *haar*)

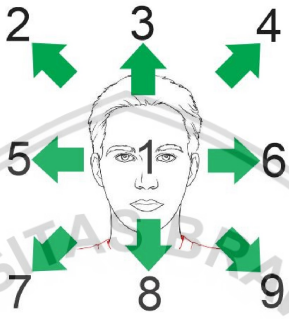
Masing-masing dari stage memiliki *threshold* yang berbeda-beda dan tiap fitur dalam stage juga memiliki *threshold* yang berbeda-beda. Objek yang tidak memenuhi akan ditolak dan yang memenuhi akan melalui *stage* selanjutnya hingga objek ditemukan.

Setelah mengetahui proses perancangan *face detection* selanjutnya ialah penulis merancang algoritma *face detection* dengan menggunakan *cascade classifier* yang sudah ada dan tersedia secara gratis di internet. Pada penelitian kali ini penulis menggunakan haarcascade_frontalface_dafault.xml yang merupakan *cascade classifier* untuk pendeteksian wajah yang dibuat oleh open cv. Algoritma *face detection* akan disajikan dalam bentuk *flowchart*. Gambar 5.7 akan menunjukan *flowchart* sistem *face detection*.



Gambar 5. 7 *Flowchart FaceDetection*

Flowchart pada gambar 5.7 sudah merangkap dalam pembuatan dataset. Penulis akan membuat dataset sebanyak 10 gambar wajah untuk masing-masing individu. Dalam flowchart, perhitungan jumlah dimulai dari 0 hingga 9 yang mana pengambilan sampel gambar akan berhenti jika variable bernilai lebih besar dari 9. Jumlah 10 ditentukan berdasarkan pertimbangan arah pandang saat melakukan pengambilan data set. Pertimbangan arah pandang wajah dapat meningkatkan akurasi pengenalan pada saat berada di beberapa kondisi tertentu. Sebagai contoh gambar 5.8 merupakan gambar kemungkinan arah pandang wajah saat menghadap ke kamera.



Gambar 5. 8 Gambar Kemungkinan Arah Pandang Wajah

Sedikit perubahan arah pandang masih dapat ditoleransi sebagai ciri wajah jika pendeteksian menggunakan haarcascade open cv. Hal ini dapat dimanfaatkan untuk meningkatkan akurasi pengenalan. Sebagai contoh simulasi, jika pengambilan data set hanya dilakukan pada 1 arah pandang saja kemungkinan wajah bisa langsung tidak dikenali ketika arah pandang bergeser sedikit. Oleh sebab itu dengan memanfaatkan 9 kemungkinan arah pandang diharapkan dapat menambah akurasi. Dari pertimbangan diatas, 10 sampel per individu dengan 9 arah pandang yang berbeda merupakan jumlah yang pas untuk pengambilan dataset.

Jumlah individu yang nantinya akan diambil sampel wajahnya berjumlah 4 orang individu. Asumsinya 4 orang individu merupakan jumlah penghuni dalam sebuah rumah yang menggunakan sistem ini. Demi memaksimalkan dalam pengujian sistem nantinya individu akan diilustrasikan sebagai berikut. Dataset terdiri dari 2 orang wanita dan 2 orang pria. Untuk wanita terdiri dari satu orang wanita berhijab dan satu orang wanita tidak berhijab. Pada persamaan 5.5 merupakan perhitungan penentuan jumlah dataset yang diperlukan untuk sistem.

$$Dataset\ Per\ Individu = jumlah\ arah\ pandang \times individu \quad \textbf{(5.5)}$$

Jumlah arah pandang diatas merupakan jumlah minimal yang dapat mewakili 9 arah berbeda. Jadi jika ditambahkan menjadi 10 untuk sampel per individu tidak menjadi masalah. Ini hanya sebagai cara tambahan agar akurasi pengenalan dapat lebih baik dibandingkan hanya satu arah saja.

Dataset Per Individu $= 10 \text{ x } 1$

$= 10$

Total Dataset $=$ Jumlah Dataset Per Individu x Jumlah Individu

$= 10 \text{ x } 4$

$= 40$

Jadi ketika sistem dioperasikan, sistem akan melakukan pengambilan dataset sebanyak 40 kali. Dan semua hasil akan disimpan dalam sebuah folder dengan nama Dataset. Masing-masing file jpg yang tersimpan akan dibedakan antara satu individu dengan individu yang lain dengan cara membedakan nama file ketika melakukan exsport data jpg. Nama file akan terdiri dari Id Wajah dan Nomor Sampel gambar. Hal ini dilakukan untuk mencegah terjadinya *replacement* data set atau tertukarnya dataset.

5.1.1.2 *Training Dataset*

Proses selanjutnya ialah mencari pola ciri wajah dari masing-masing individu untuk membedakan antara individu satu dengan yang lainnya. Dibutuhkan operator untuk klasifikasi tekstur dan pada penelitian kali ini penulis menggunakan *Local Binary Patern*. Secara teori *local binary pattern* adalah salah satu deskriptor tekstur terbaik dan telah banyak digunakan dalam berbagai aplikasi. Cara kerja nya ialah dengan mengkonversi seluruh citra *grayscale* yang telah dijadikan *dataset* sebelumnya menjadi *Local Binary Patern Image* dan dijadikan sebuah *Histogram*. Berikut merupakan langkah-langkah membuat *Local Binary Patern*.

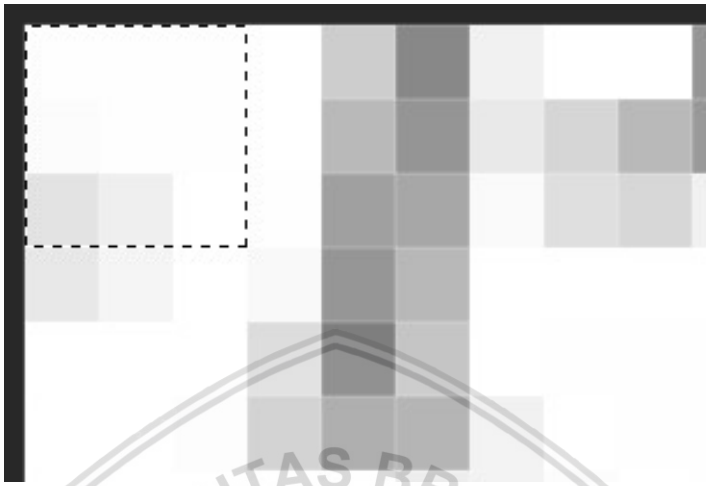
1. Siapkan citra *grayscale* yang akan diubah menjadi *local binary pattern image*. Gambar 5.9 merupakan citra *Grayscale*.

Gambar 5. 9 Citra *GrayScale*

2. Menggunakan rumus LBP 3 x 3 hitunglah mulai piksel pojok kiri atas. Rumus yang digunakan diletakan pada persamaan 5.6.

$$LBP_{P.R} = \sum_{p=0}^{p-1} s\,(g_p - g_c)\,2^p \qquad s\,(x) = \begin{cases} 1, if\ x \geq 0; \\ 0, otherwise. \end{cases} \qquad \textbf{(5.6)}$$

3. Setelah dilakukan *zoom* hingga terlihat jelas tiap piksel ambil matrik 3 x 3 pada piksel, maka gambar akan menjadi seperti pada gambar 5.10.



Gambar 5. 10 Matrik 3 x 3 Pojok Kiri Atas Citra

4. Hitung nilai matrik 3 x 3 yang telah ditandai tersebut menggunakan rumus LBP. Nilai matriknya adalah sebagai berikut:

$$\begin{matrix} 254 & 254 & 255 \\ 239 & 255 & 254 \\ 245 & 255 & 249 \end{matrix}$$

Titik pusatnya ialah 255 kemudian bandingkan dengan matrik disekitarnya dengan aturan jika tittik pusat ≥ nilai piksel sekitar maka beri nilai 0. Jika titik pusat < nilai piksel sekitar maka beri nilai 1. Setelah dilakukan seperti itu kemudian matriknya akan menjadi seperti berikut :

$$\begin{matrix} 0 & 0 & 1 \\ 0 & Pusat & 0 \\ 0 & 1 & 0 \end{matrix}$$

Untuk mengetahui nilai pusat, hasil biner tersebut kemudian diubah menjadi desimal dengan cara seperti berikut :

$$\begin{matrix} 0_7 & 0_6 & 1_5 \\ 0_0 & Pusat & 0_4 \\ 0_1 & 1_2 & 0_3 \end{matrix}$$

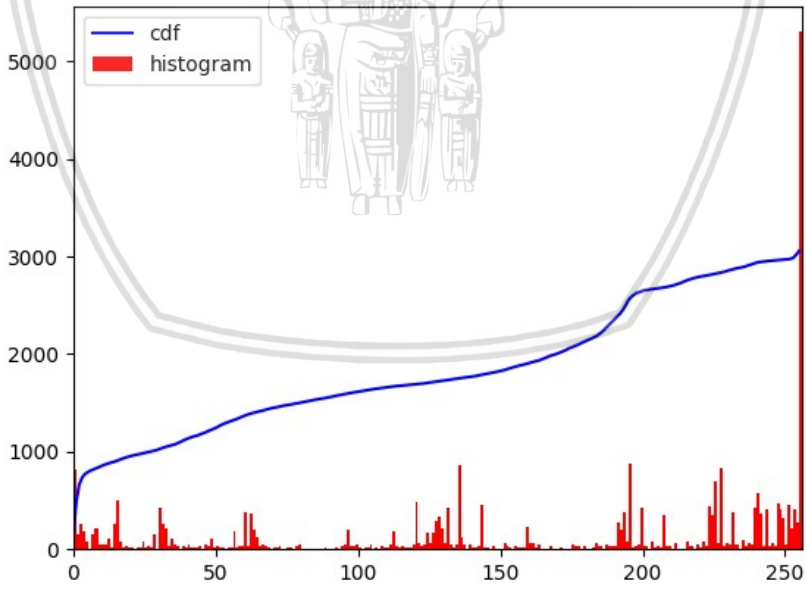
$$0_7\ 0_6\ 1_5\ 0_4\ 0_3\ 1_2\ 0_1\ 0_0$$
$$2^5 \qquad 2^2$$

0 +0 +32+0+0 +4 +0 +0 = 36

Dari hasil perhitungan tersebut diketahui nilai pusat nya ialah 36. Jika perhitungan diimplementasikan pada semua bagian citra maka citra LBP akan menjadi seperti pada gambar 5.11.

Gambar 5. 11 LBP image

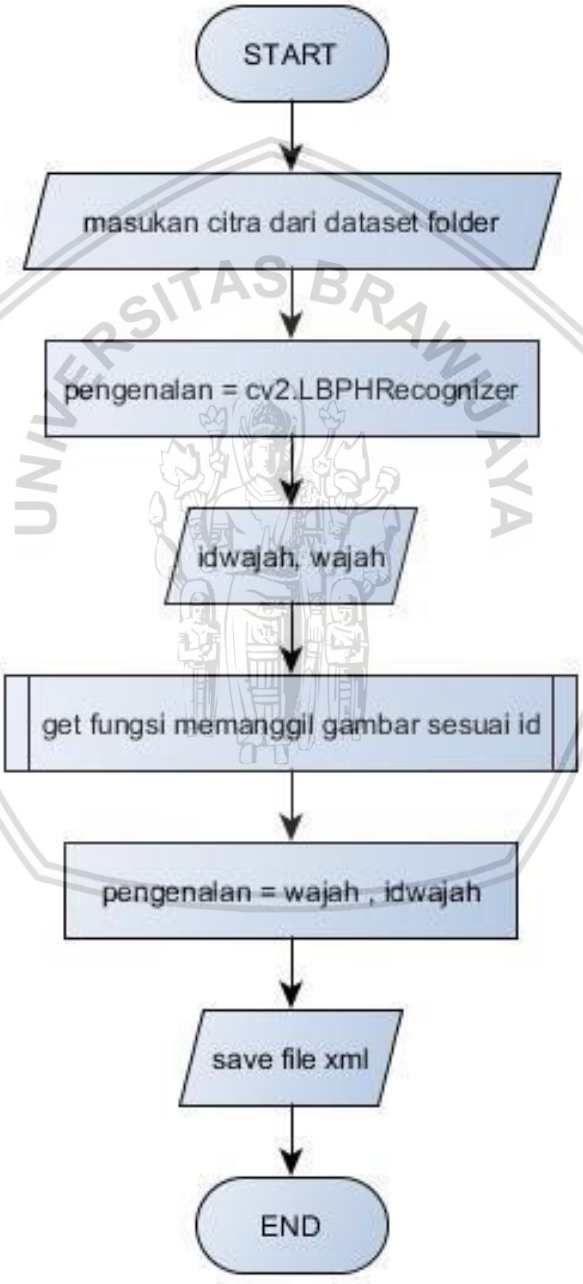
Dan hasil perhitungan LBP jika disajikan dalam bentuk Histogram akan menjadi seperti pada gambar 5.12 dibawah ini :



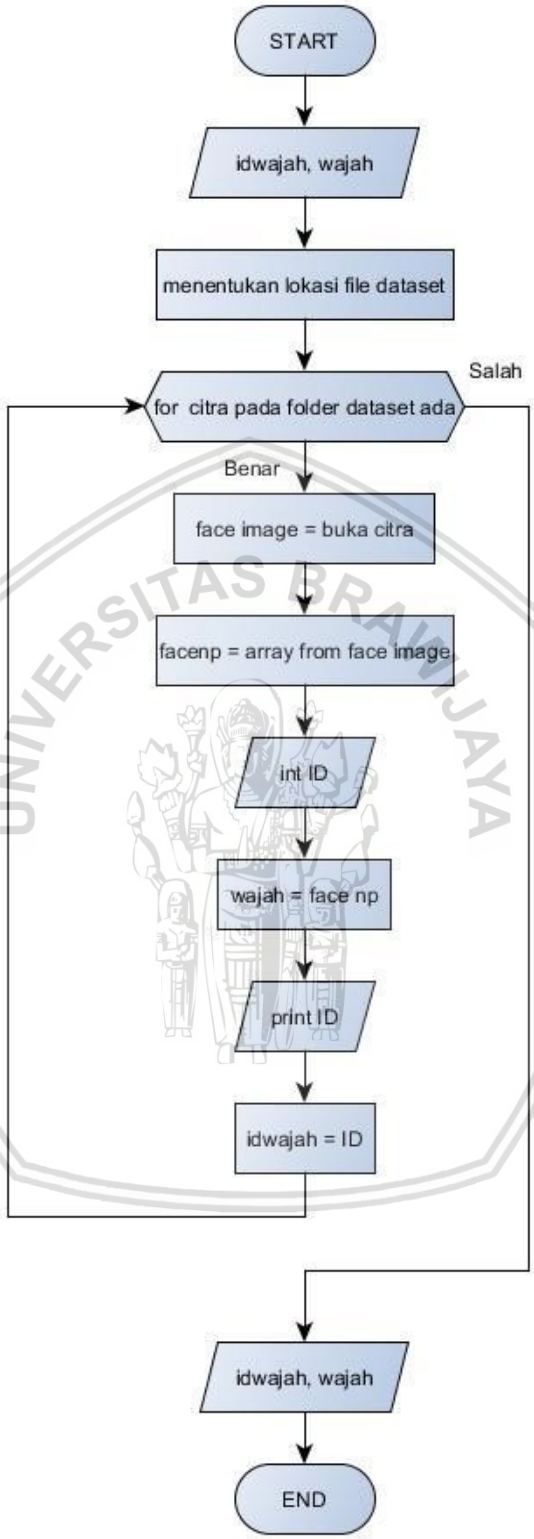
Gambar 5. 12 *Local Binary Pattern Histogram*

Kemudian setiap hasil tersebut akan disimpan dalam sebuah file xml dan juga dari hasil tersebut akan diklasifikasikan tekstur wajah untuk selanjutnya dijadikan tolak ukur perbandingan citra yang akan menjadi masukan selanjutnya.

Setelah melalui proses tersebut selanjutnya penulis akan membuat algoritma yang dapat menyimpan hasil LBP dalam bentuk file xml. Algoritma akan dirancang dalam bentuk *Flowchart*. Untuk Fungsi *Local Binary Patern Histogram* penulis menggunakan *Library* buatan *Open CV* yaitu cv2.createLBPHFaceRecognizer. *Library* tersebut sudah mencangkup keseluruhan cara kerja *Local Binary Pattern*. Pada gambar 5.13 dan gambar 5.14 akan dijelaskan *flowchart code* program yang akan dibuat oleh penulis.



Gambar 5. 13 *Flowchart Main Program* Training Data

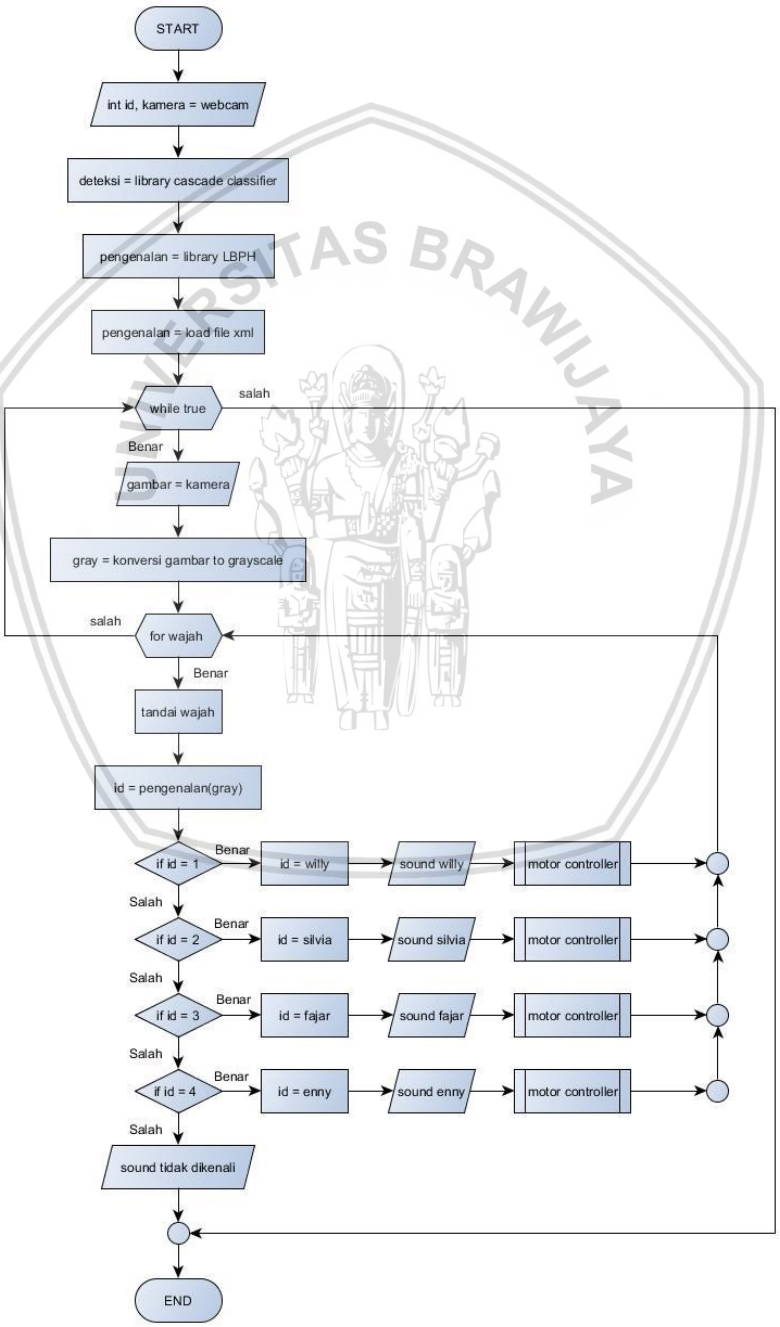


Gambar 5. 14 *Flowchart* Fungsi *Load Image* Sesuai ID

Hasil dari training data ialah file xml dengan data LBPH masing-masing citra pada folder dataset yang telah di training.

5.1.1.3 *Face Recognition*

Setelah melakukan proses training yang selanjutnya ialah me-*load* hasil training yaitu file xml yang telah disimpan sebelumnya dan dibandingkan dengan *face detection* baru akan dilakukan. Fase ini disebut *face recognition* yang berarti pengenalan wajah dimana data wajah sudah disimpan sebelumnya kemudian dibandingkan dengan data yang baru. Jika *match* maka wajah akan dikenali sebagai seseorang dengan cara menambahkan *variable* nama pemilik wajah. Pada gambar 5.15 berikut adalah *flowchart* dari perancangan perangkat lunak :



Gambar 5. 15 *Flowchart Face Recognition*

5.1.1.4 Motor Kontroller

Selanjutnya ialah merancang bagaimana caranya menggerakan motor. Menggerakan motor pada setiap mikrokontroller memiliki sedikit perbedaan. Pada raspberry pi penggunaan Pin GPIO untuk menggerakan motor adalah cara yang paling efektif. Dengan tambahan berupa *driver* motor dapat mempermudah mengontrol motor bahkan jika menggunakan lebih dari 1 buah motor. Gambar 5.16 merupakan *flowchart* program untuk menggerakan motor dc yang sebelumnya dihubungkan melalui IC L293D dan raspberry pi.

Gambar 5. 16 Flowchart Motor *Controller*

Sebelum membahas *flowchart* lebih jauh, perlu diketahui bahwa ada tabel untuk pergerakan motor menggunakan IC L293d. Tabel 5.1 akan menunjukan pergerakan motor pada IC L293D:

Tabel 5. 1 Tabel Motor *Controller*

Enable	**A**	**B**	**Hasil**
Low	*High*	*Low*	Tidak berputar
Low	*Low*	*High*	Tidak Berputar
High	*Low*	*Low*	Tidak Berputar
High	*High*	*Low*	Berputar Searah Jarum Jam
High	*Low*	*High*	Berputar Berlawanan Arah Jarum Jam
High	*HIgh*	*High*	Tidak Berputar

Pertama-tama inisialisasi pin berapa saja pada raspberry pi yang akan digunakan untuk masukan motor. Harus menggunakan pin GPIO untuk masukan pada motor karena motor tidak akan bekerja jika menggunakan pin selain GPIO. Dibutuhkan 3 buah masukan untuk satu buah motor. Karena sistem menggunakan 3 buah motor, namun sistem hanya membutuhkan 6 buah masukan karena motor 2 dan 3 berada pada pin yang sama pada IC L293D. Pada dasarnya pin GPIO ialah pin tegangan yang akan bernilai 0 volt jika tegangan *low* dan akan bernilai 5 volt jika tegangan *high*.

Penulis menggunakan pin 16 dan 18 untuk masukan tegangan motor 1 dan pin 12 untuk *enable* motor 1 atau yang bertugas sebagai *switch on/off* pada motor 1. Pada *flowchart*, dibutuhkan waktu 1 detik untuk memutar motor 1, lebih singkat jika dibanding dengan motor 2. Hal ini dikarenakan motor 1 hanya bertugas untuk menutup dan membuka kunci. Kunci didesain seperti kail yang hanya membutuhkan pergerakan motor sebanyak 180 derajat untuk menggerakan kunci ke posisi buka dan tutup. Pada tabel 5.1 dibutuhkan tegangan high dan low serta enable dalam posisi high untuk memutar motor searah jarum jam. Motor akan diposisikan agar berputar searah jarum jam untuk membuka kunci dan melawan arah jarum jam untuk menutup kunci. Pada *flowchart*, penulis memanfaatkan kondisi motor saat mati untuk memberi jeda antar proses. Hal ini diberikan agar motor pada proses selanjutnya tidak langsung bergerak.

Kemudian Penulis menggunakan pin 38 dan 40 untuk masukan tegangan motor 2 dan 3 dan pin 36 untuk enable motor 2 dan 3 atau yang bertugas sebagai switch on/off pada motor 2 dan 3. Pada flowchart, dibutuhkan waktu 5 detik untuk memutar motor 2 dan 3, lebih lama jika dibanding dengan motor 1. Hal ini dikarenakan motor 2 dan 3 bertugas untuk menutup dan membuka pintu. Pintu didesain agar dapat digeser dan hal ini membutuhkan beberapa rotasi roda untuk membukanya secara sempurna. Oleh sebab itu diperlukan waktu kurang lebih 5 detik untuk bisa terbuka sempurna. Pada tabel 5.1 dibutuhkan tegangan high dan low serta enable dalam posisi high untuk memutar motor searah jarum jam. Motor akan diposisikan agar berputar searah jarum jam untuk membuka pintu dan melawan arah jarum jam untuk menutup.

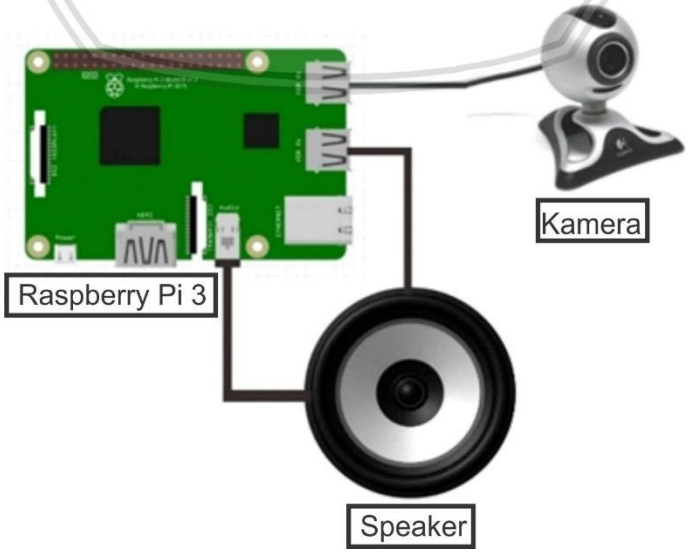
Motor 2 dan 3 akan diletakan pada bagian bawah pintu. Motor akan diberikan dudukan motor agar dapat menggunakan ban dari *line tracer*. Motor 2 akan menghandle dua buah ban disebelah kiri dan kanan. Begitu juga untuk motor 3 akan menghandle dua buah ban disebelah kiri dan kanan. Penggunaan 4 buah roda sebagai penggerak bertujuan agar pintu dapat seimbang ketika bergerak dan juga pintu benar-benar dapat bergerak dengan lancer. Pada perancangan sebelumnya penulis menggunakan gear untuk menggerakan pintu dengan bantuan 1 buah motor. Namun ketika sampai pada fase pengujian, pintu tidak dapat bergerak atau terbuka sama sekali, perlu diberikan sedikit pelumas dan didorong untuk pergerakan awal agar pintu dapat terbuka. Berdasarkan pertimbangan tersebut, penggunaan ban line tracer sebagai ban pada pintu merupakan cara yang paling efektif dan efisien agar pintu dapat bergerak sempurna.

5.1.2 Perancangan Perangkat Keras

Perancangan perangkat keras dilakukan melalui beberapa tahap dimulai dari tahap membuat rangkaian hingga prototipe. Perancangan yang pertama ialah Rangkaian Webcam, Raspberry pi dan Speaker dilanjutkan Rangkaian Motor DC dengan IC L293D, Rangkaian Raspberry pi dengan IC L293D, Power Supply Raspberry pi, IC dan Motor DC dan yang terakhir ialah Desain prototipe pintu.

5.1.2.1 Rangkaian *Webcam*, Raspberry pi dan *Speaker*

Webcam dan *speaker* merupakan komponen yang tidak menggunakan jumper untuk dihubungkan dengan keseluruhan sistem. Untuk perancangan sistem ini, *Webcam* dan *Speaker* akan memanfaatkan *port* USB yang ada pada *Raspberry pi* untuk menghubungkannya dengan sistem. Hal ini dikarenakan sudah banyaknya *speaker* dan *webcam* yang menggunakan *port* USB. Namun untuk *speaker* akan menggunakan tambahan *port* yaitu dengan memanfaatkan *port* Jack audio 3,5 mm yang ada pada raspberry pi 3. Pada gambar 5.17 akan digambarkan desain hubungan antara webcam, raspberry pi dan speaker secara keseluruhan.



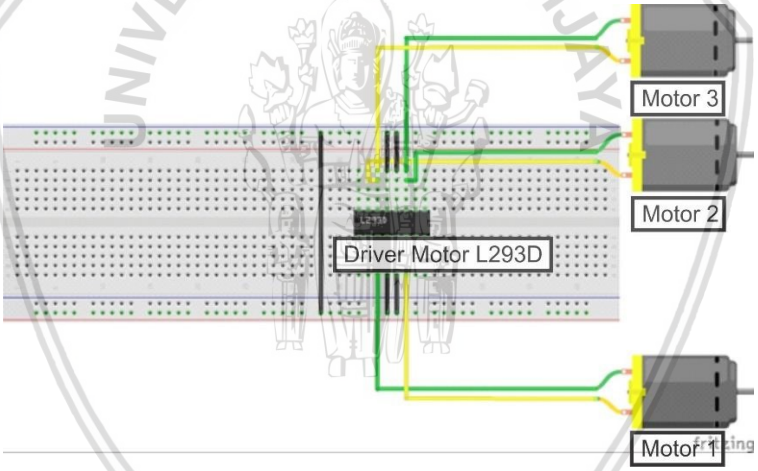
Gambar 5. 17 Desain Perancangan Raspberry pi, *Speaker* dan *Webcam*

Raspberry pi memiliki 4 *port* USB 2.0 dengan memanfaatkan salah satunya webcam dapat langsung dihubungkan melalui salah satu *port* usb 2.0 . Data yang dikirimkan haruslah data serial. Perlu diketahui juga bahwa usb 2.0 memiliki 4 kabel penghubung yang terdiri dari VCC, D+, D- dan juga GROUND. Jadi usb juga sudah bisa mengcover *power* untuk *webcam*. Penggunaan usb dirasa paling sederhana dan paling efektif sehingga penulis memilih untuk menggunakan *port* usb untuk memenuhi kebutuhan *power web cam*.

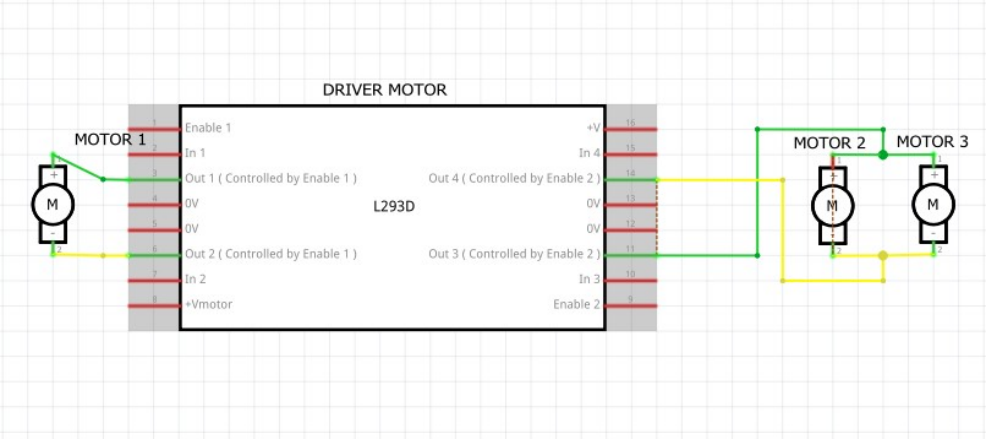
Untuk speaker, penulis memanfaatkan *port* usb pada raspberry pi namun untuk speaker ada tambahan masukan yaitu melalui *Jack Audio*. Perlu diketahui juga *jack audio* yang digunakan ialah 3,5 mm *jack audio* dengan 3 komunikasi data yaitu *speaker left*, *speaker right* dan *ground*. Untuk *power* pada speaker dikirimkan melalui USB *port* dengan memanfaatkan +5V vcc yang ada pada USB *port.*

5.1.2.2 Rangkaian Motor DC dengan IC L293D

Pembuatan rangkaian sistem khususnya hubungan antara motor DC dengan IC L293D akan dilakukan pada sebuah *project board* dengan *jumper* sebagai penghubungnya. Gambar 5.18 dan gambar 5.19 merupakan desain rangkaian dan juga desain *schemantic* rangkaian.



Gambar 5. 18 Gambar Desain Rangkaian IC L293D dan Motor DC

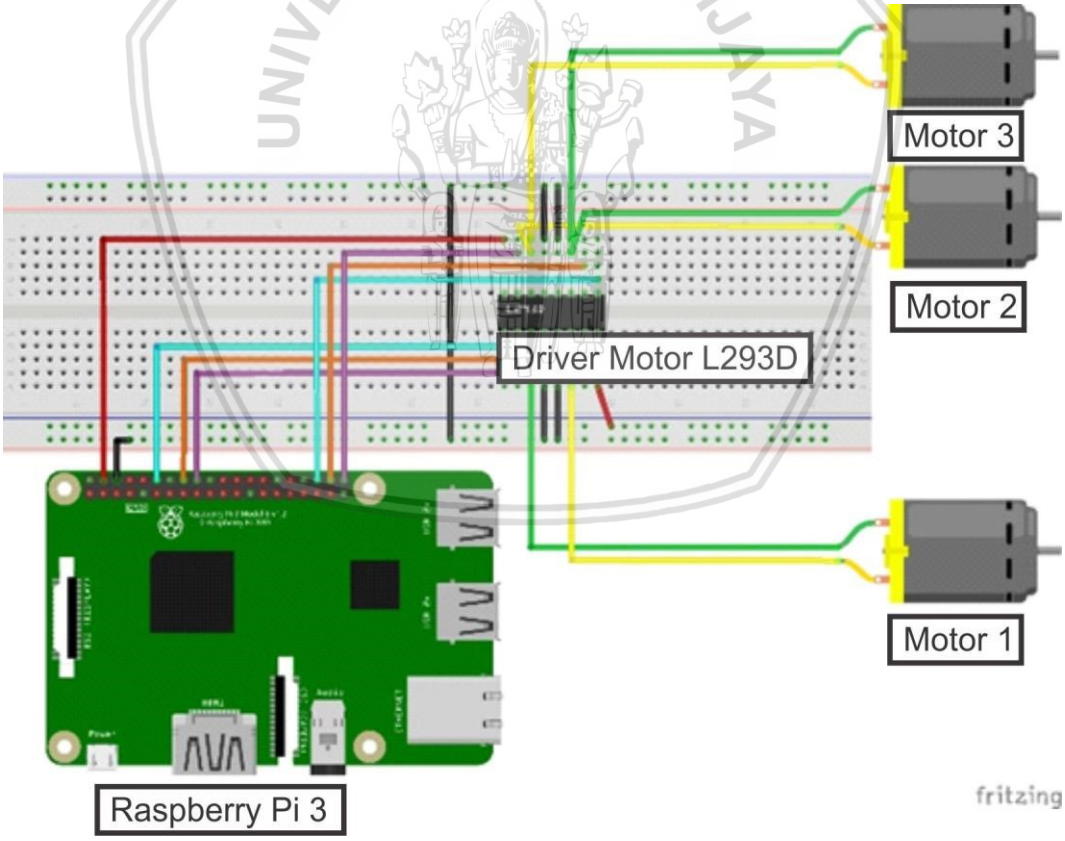


Gambar 5. 19 Desain *Schemantic* Rangkaian

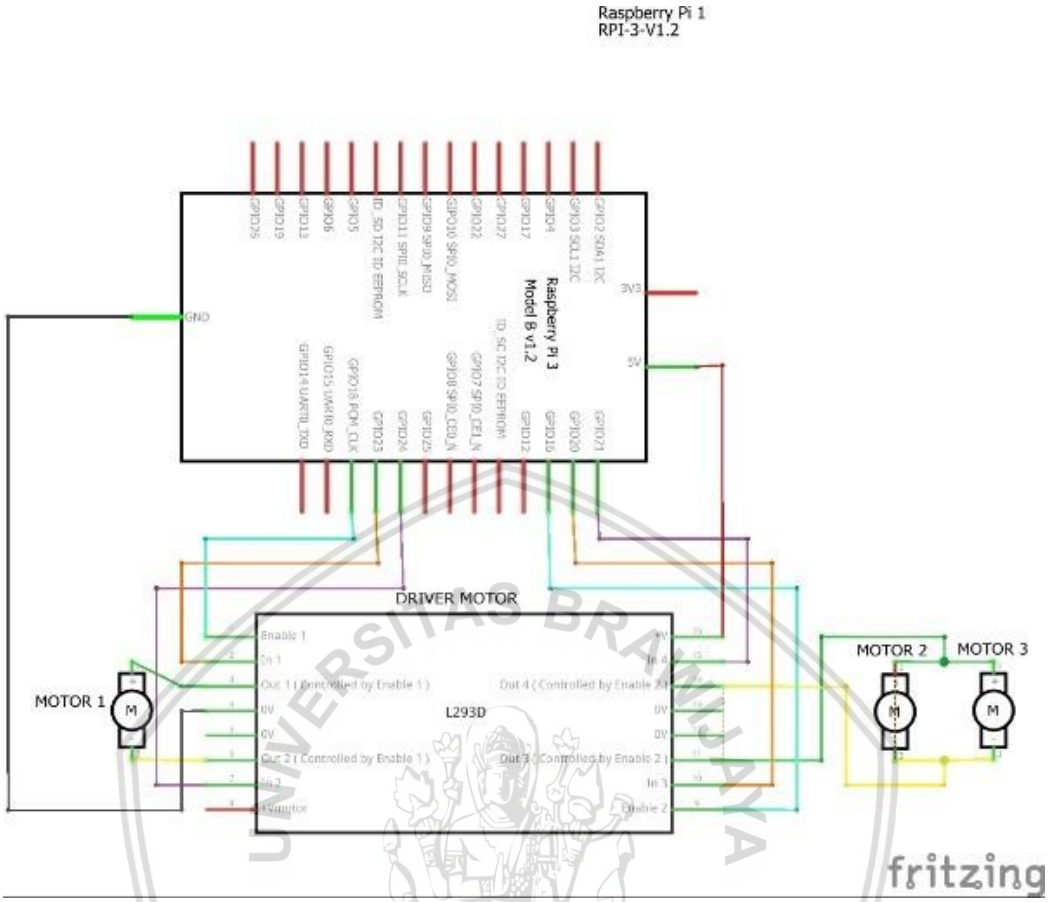
Dari gambar 5.20, penulis hanya menghubungkan 3 buah motor DC yang dibutuhkan pada pin yang sesuai. Motor DC yang digunakan disini merupakan motor DC 6V jadi untuk menggerakkan motor secara maksimal akan dibutuhkan tegangan DC minimal 6V. Motor 1 nantinya yang akan bertugas sebagai pembuka dan pengunci pintu. Motor 1 dihubungkan pada pin 3 dan pin 6 pada IC yang mana nantinya ON/OFF nya dapat diatur melalui Pin 1 Atau Enable 1. Untuk motor 2 dan 3 akan bertugas sebagai penggerak buka tutup pintu. Motor 2 dan 3 akan bertindak sebagai roda dari pintu dan akan dihubungkan pada pin 11 dan 14 dan *enable* dilakukan oleh pin 9. Kemudian pin 4, 5, 12, 13 akan dihubungkan ke ground sistem yang nantinya akan dihubungkan dengan ground raspberry pi 3.

5.1.2.3 Rangkaian Raspberry pi dengan IC L293D

Perancangan selanjutnya ialah komunikasi antara raspberry pi dengan IC L293D. Komunikasi nantinya akan menggunakan GPIO pin pada raspberry pi. Pin GPIO merupakan pin yang sangat tepat digunakan untuk mengontrol motor dengan Raspberry pi. Gambar 5.20 dan gambar 5.21 akan menjelaskan rangkain raspberry pi dan IC L293D dalam bentuk desain rangkaian dan juga desain schemantic.



Gambar 5. 20 Gambar Desain Rangkaian Raspberry Pi dan IC L293D



Gambar 5. 21 Gambar Desain Schemantic Rangkaian Raspberry Pi dan IC L293D

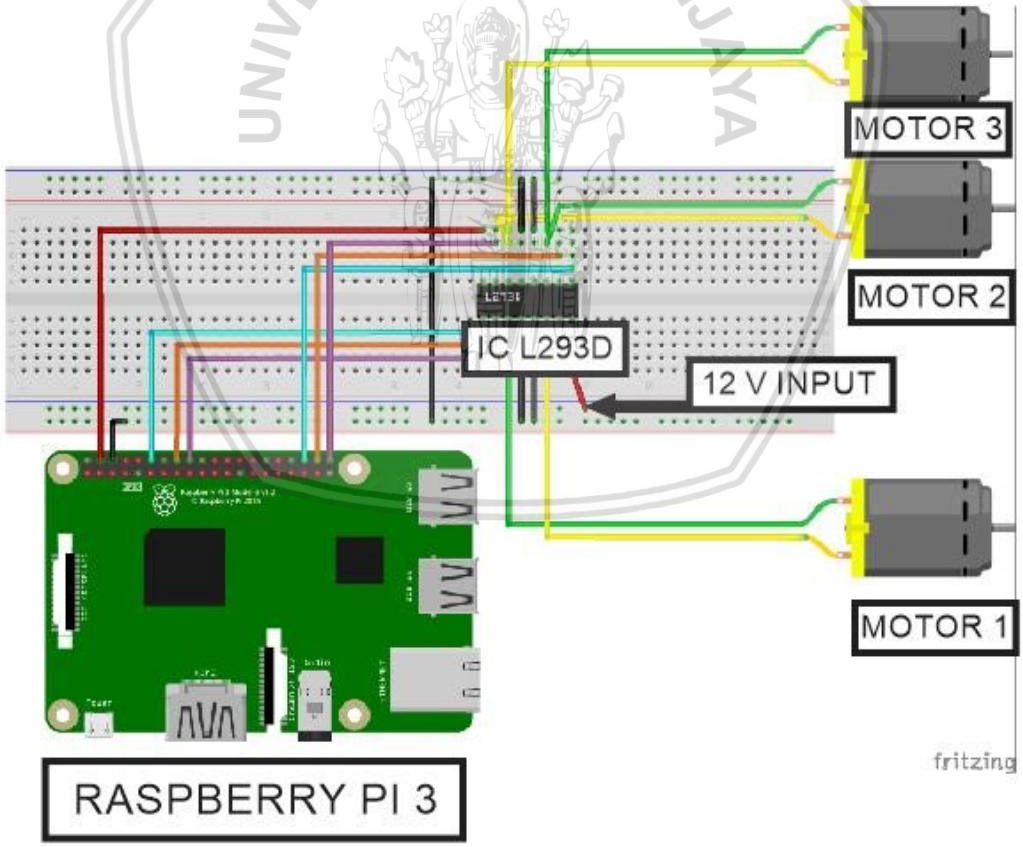
Pada rangkaian diatas penulis merancang menggunakan Pin GPIO pada Raspberry pi. GPIO merupakan singkatan dari *General Purpose Input Output* yang berarti pin yang digunakan untuk mengatur input maupun output dengan cara bertindak sebagai trigger tegangan output yang akan bernilai 5V ketika aktif dan akan bernilai 0V ketika *non*-aktif. GPIO dapat diinisialisasikan pada coding Raspberry pi dengan menyebutkan pin brapa saja yang digunakan.

Untuk mengatur motor 1 yang akan berfungsi sebagai kunci pada pintu, GPIO 18 akan dihubungkan dengan IC L293D pada Pin 1. Komunikasi ini akan berfungsi sebagai pengatur On/Off motor 1 dengan cara memberikan tegangan High ketika ingin mengaktifkan motor dan tegangan low ketika ingin menon-aktifkan motor 1. Kemudian GPIO 23 dihubungkan dengan Pin 2 IC L293D dan GPIO 24 dihubungkan dengan pin 7 IC L293D. Dua komunikasi ini akan mengatur arah putaran motor 1. Ketika Pin 2 Diberikan tegangan High dan pin 7 diberikan tegangan Low maka motor DC 1 akan berputar searah jarum jam kunci akan terbuka dan ketika Pin 2 diberikan tegangan Low dan pin 7 diberikan tegangan High maka motor DC 1 akan berputar berlawanan arah jarum jam. Hal ini diperlukan untuk sistem karena motor DC 1 bertugas sebagai pengunci dan pembuka kunci pintu sehingga dibutuhkan putaran bolak-balik pada motor DC 1.

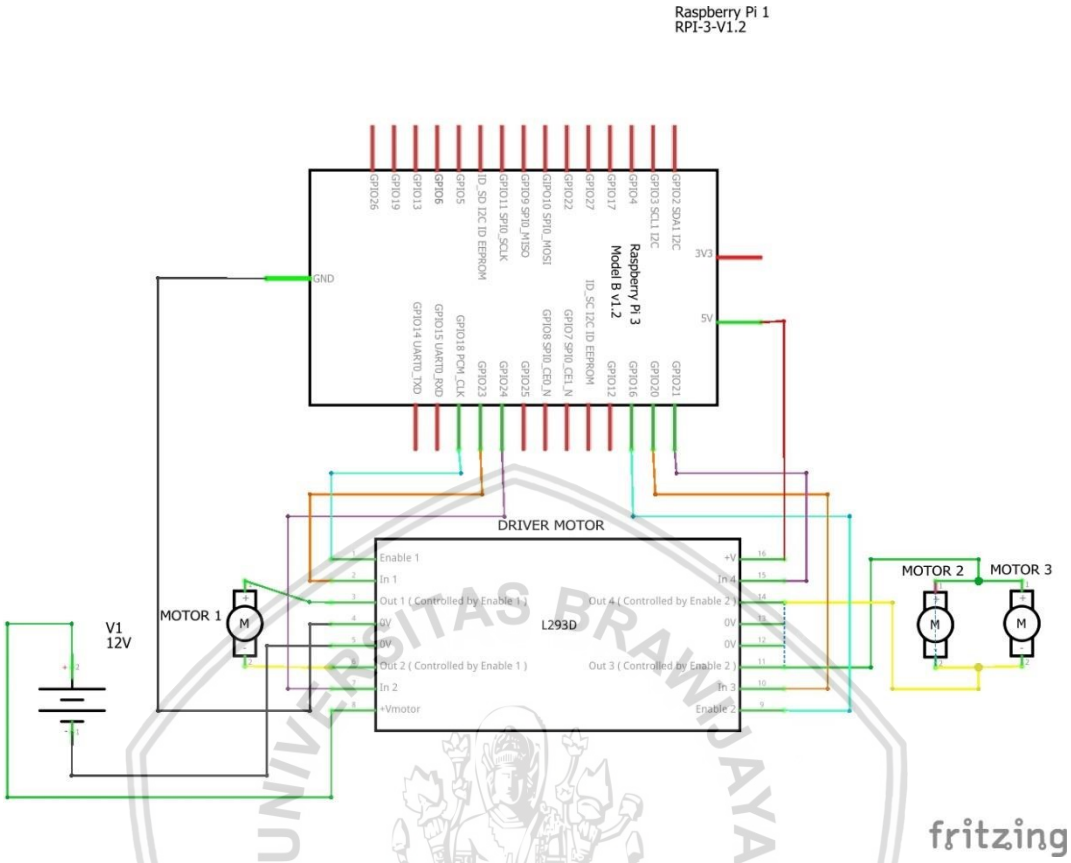
Untuk mengatur motor 2 dan 3, GPIO 16 akan dihubungkan dengan IC L293D pada Pin 9. Komunikasi ini akan berfungsi sebagai pengatur On/Off motor 2 dan 3 dengan cara memberikan tegangan High ketika ingin mengaktifkan motor dan tegangan low ketika ingin menon-aktifkan motor. Kemudian GPIO 20 dihubungkan dengan Pin 10 IC L293D dan GPIO 21 dihubungkan dengan pin 15 IC L293D. Dua komunikasi ini akan mengatur arah putaran motor. Ketika Pin 10 Diberikan tegangan High dan pin 15 diberikan tegangan Low maka motor akan berputar searah jarum jam dan ketika Pin 10 diberikan tegangan Low dan pin 15 diberikan tegangan High maka motor akan berputar berlawanan arah jarum jam. Hal ini diperlukan untuk sistem karena motor 2 dan 3 bertugas sebagai pembuka dan penutup pintu, jadi butuh putaran bolak-balik pada motor 2 dan 3.

5.1.2.4 Power Supply Raspberry pi, IC dan Motor DC

Pada perancangan power supply dibutuhkan dua buah power supply untuk sistem yaitu power supply untuk raspberry pi dan juga power supply untuk motor. Gambar 5.22 dan gambar 5.23 akan menjelaskan desain rangkaian power supply raspberry pi, IC dan motor.



Gambar 5. 22 Gambar Desain Rangkaian Power Supply Raspberry pi, IC dan Motor DC

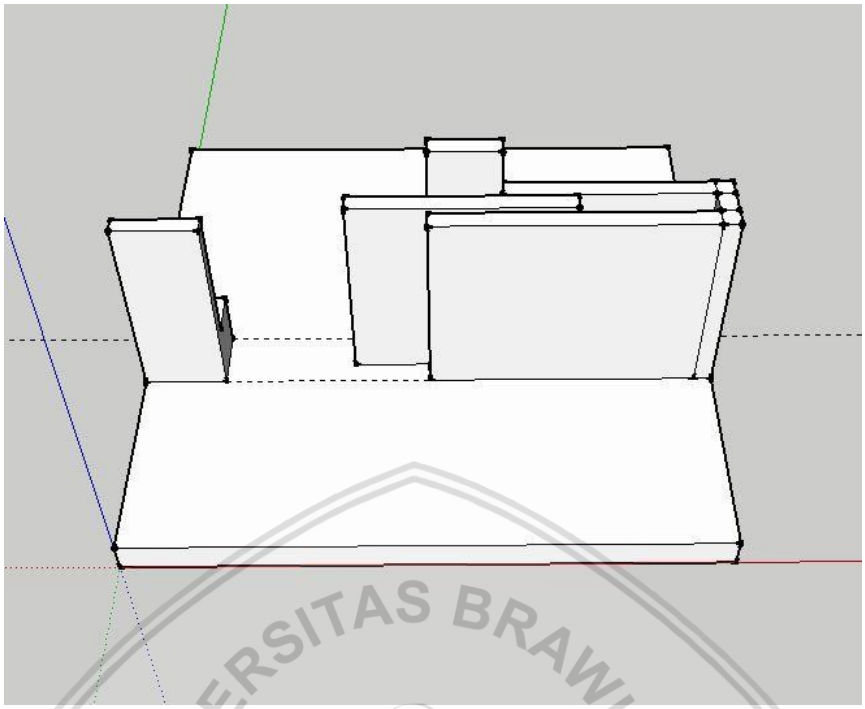


Gambar 5. 23 Gambar Desain Schemantic Power Supply Raspberry pi, IC dan Motor DC

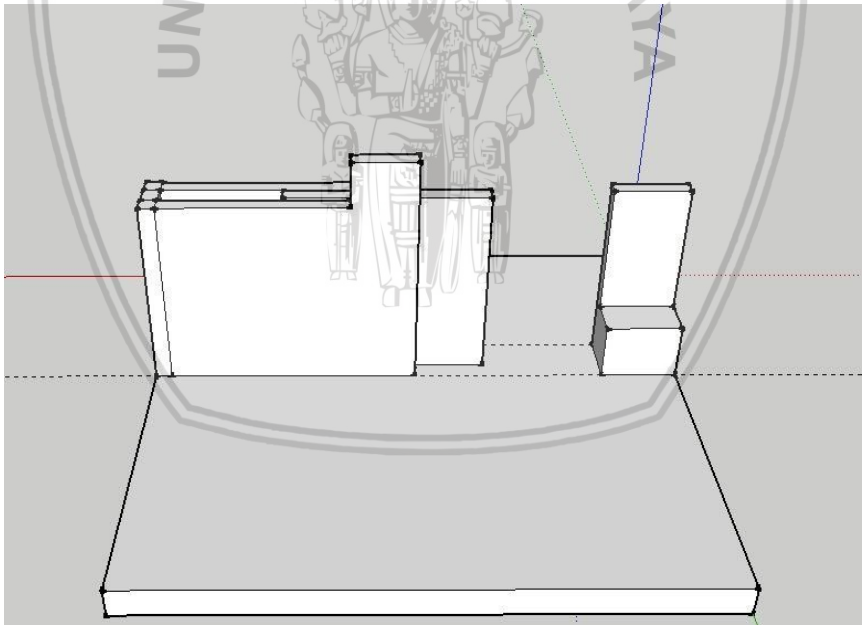
Sistem akan menggunakan power supply berupa adapter 12v untuk motor DC dan juga adapter 5.1V 2.5A untuk raspberry pi. Penggunaan power supply untuk raspberry pi harus sesuai dengan standar yang telah ditentukan oleh raspberry. hal ini dilakukan supaya raspberry pi dapat bekerja dengan maksimal dan tidak kekurangan daya. Dapat dilihat pada situs *Raspberrypi.org* bahwa raspberry pi 3 akan bekerja dengan maksimal pada tegangan 5.1 v dan arus 2.5A. Dan untuk power supply pada motor DC, Penulis menggunakan adapter 12v yang nantinya akan disambungkan terlebih dahulu kedalam IC L293D yang kemudian akan diteruskan ke motor. Penggunaan adapter 12v adalah power yang effisien untuk diterima oleh ICL293D.

5.1.2.5 Desain Prototipe Miniatur Pintu

Pada tahap ini penulis merancang desain 3D menggunakan google sketch up untuk membuat prototipe berupa miniature simulasi pintu rumah. Pada gambar 5.24 dan gambar 5.25 adalah desain prototipe tampak depan dan belakang.



Gambar 5. 24 Gambar Desain Tampak Depan

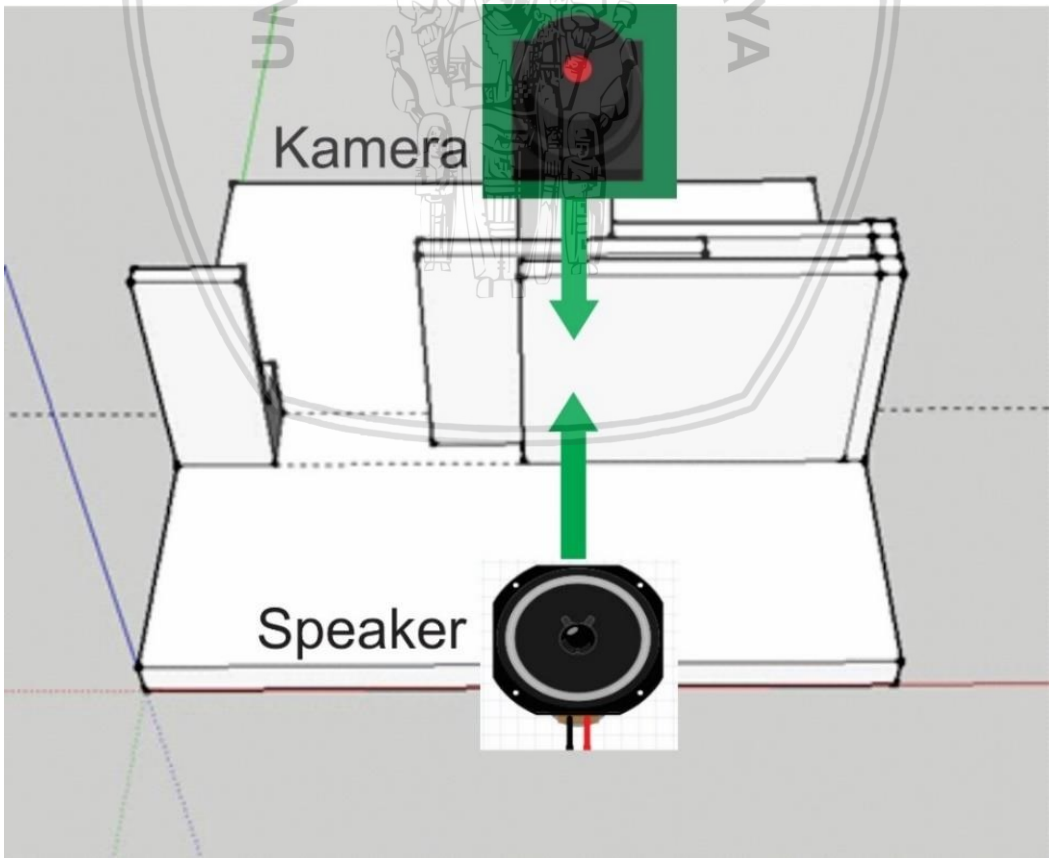


Gambar 5. 25 Gambar Desain Tampak Belakang

Sistem nantinya akan dibuat dalam bentuk prototipe miniatur pintu. Pada gambar diatas kita dapat melihat bahwa pintu nantinya akan dibuat bergeser. Hal ini dikarenakan penulis menggunakan motor untuk membuka dan menutup pintu. Bahan yang akan digunakan dalam pembuatan prototipe ialah berupa bahan yang biasa digunakan untuk membuat maket pada bidang keilmuan arsitektur yaitu kertas PVC.

Untuk sistem membuka dan menutup pintu, penulis menggunakan 4 buah roda yang akan di letakan pada bagian bawah pintu. Motor 2 dan 3 nantinya akan menghandle masing-masing 2 buah roda. Menggunakan 4 buah roda merupakan cara yang paling efektif agar pintu dapat bergeser dan jumlah 4 roda mempertimbangkan faktor keseimbangan pintu pada prototipe. Roda yang digunakan ialah roda yang biasa dipakai pada robot *Line Tracer*. Kemudian, untuk sistem membuka dan mengunci pintu, motor 1 akan diletakan pada bagian pojok pintu. Pada motor akan diletakan sebuah kail untuk mengunci pintu dengan memanfaatkan perputaran motor dan mengaitkannya pada penyangga yang telah diletakan pada bagian ujung pintu.

Untuk peletakan rangkaian nantinya akan diletakan pada bagian dalam prototipe pintu, hal ini dilakukan agar manajemen kabel jumper tidak terlalu terlihat acak-acakan jika dilihat dari bagian depan pintu. Dan untuk bagian depan pintu nantinya akan diletakan kamera dan speaker yang akan dimasukan dalam sebuah box yang telah dilubangi sebelumnya dan pengguna nantinya dapat meletakan box tersebut disamping pintu bagian depan rumah jika sudah diimplementasikan pada sebuah rumah yang sesungguhnya. Agar lebih jelas gambar 5.26 berikut merupakan gambar peletekan kamera dan speaker.



Gambar 5. 26 Peletakan Kamera dan *Speaker*

5.2 Implementasi Sistem

Pada sub bab ini menjelaskan tentang implementasi sistem berdasarkan perancangan yang sudah dibuat pada bagian sebelumnya. Pada implementasi sistem terdapat implementasi perangkat keras dan implementasi perangkat lunak.

5.2.1 Implementasi Perangkat Lunak

5.2.1.1 *Face Detection*

Sudah disebutkan bahwa, proses implementasi untuk face recognition dibagi menjadi tiga bagian. Bagian-bagian tersebut ialah pembuatan *Face Detection*, *Training Dataset* set dan terakhir ialah *face recognition*. Yang pertama ialah implementasi *face detection* dalam sebuah *code program*. Semua kode program ditulis menggunakan bahasa pemrograman Python hal ini dilakukan karena python ialah bahasa yang digunakan pada Raspberry pi 3. Tabel 5.2 merupakan *code* program untuk *face detection* sekaligus *create dataset* untuk disimpan dalam sebuah folder.

Tabel 5. 2 Kode Program Create Dataset

No	*Source Code*
1	`import cv2`
2	`import numpy as np`
3	
4	`deteksi =`
5	`cv2.CascadeClassifier('haarcascade_frontalface_de`
6	`fault.xml')`
7	`kamera = cv2.VideoCapture(0)`
8	`idwajah = raw_input('Masukan Nomor ID Wajah : ')`
9	`nomorsampel = 0;`
10	
11	
12	`while(True):`
13	`    ret, gambar = kamera.read();`
14	`    gray = cv2.cvtColor(gambar,`
15	`cv2.COLOR_BGR2GRAY)`
16	`    wajah = deteksi.detectMultiScale(`
17	`    gray,`
18	`    scaleFactor=1.1,`

No	*Source Code*
19	`    minNeighbors=5,`
20	`    minSize=(30, 30),`
21	`    flags = cv2.cv.CV_HAAR_SCALE_IMAGE`
22	`    )`
23	
24	`    for (x,y,w,h) in wajah:`
25	`        nomorsampel=nomorsampel+1;`
26	`        cv2.imwrite("Data_Wajah/Wajah."+idwajah`
27	`    +'.'+str(nomorsampel) + ".jpg",`
28	`    gray[y:y+h,x:x+w])`
29	`    if (nomorsampel>9):`
30	`        break`
31	
32	`kamera.release()`
33	`cv2.destroyAllWindows()`

Pembuatan dataset ialah proses dimana program melakukan *Haar like features* secara masal yang menggunakan *cascade clasiffier*. Penjelasan tabel 5.2 diatas ialah sebagai berikut:

Pada baris pertama dan kedua ialah inisialisasi library yang digunakan yaitu open cv dan juga numpy. Penggunaan library open cv bertujuan untuk penggunaan *cascade classifier* dan juga beberapa fungsi lainnya yang tersedia dalam open cv dan kebanyakan dari fungsi tersebut berguna untuk melakukan operasi di bidang *image processing*. Penggunaan Numpy disini ialah untuk melakukan perhitungan matematika terutama pengoprasian array.

Pada baris ke-4 hingga baris ke-9 merupakan proses inisialisasi variable yang akan digunakan dalam *code* program. Variabel deteksi akan berisi lokasi dari *file cascade classifier* yang digunakan. Variabel kamera berisi izin mengakses *webcam* untuk melakukan pengambilan video. Selanjutnya ialah ada Nomor sample dan ID wajah yang dalam hal ini berfungsi sebagai pemberi ciri ketika melakukan perulangan ketika melakukan pembuatan dataset.

Pada baris ke-12 hingga baris ke-30 merupakan proses dimana program melakukan pengambilan data berupa gambar yang di*capture* dengan *webcam*. Pertama-tama kamera akan mengambil gambar untuk syntax ialah yang berada pada baris ke-13. Selanjutnya pada baris ke-14 dilakukan konversi gambar ke *gray-scale* dengan memanfaatkan fungsi konversi warna *from rgb to grayscale* yang

dimiliki oleh open cv. Pada baris ke-16, dilakukaan proses pendeteksian *object* dengan menggunakan *cascade classifier* yang telah diinisialisasi sebelumnya. Proses *detectMultiScale* membutuhkan masukan berupa gambar yang berisi object yang akan dideteksi dan sudah di konversi ke *grayscale*. Dan pada proses ini keluarannya ialah gambar dengan bentuk persegi dan berisi hanya object yang terdeteksi saja, dimana dalam hal ini objek yang terdeteksi adalah wajah. Selanjutnya pada baris 24 hingga 30 ialah proses pemberian ciri atau nama pada setiap gambar hasil keluaran proses sebelumnya dan menyimpannya pada sebuah folder yang diberi nama Data_Wajah.

5.2.1.2 Training *Dataset*

Tabel 5. 3 Kode Program Training *Dataset*

No	Source Code
1	`import os`
2	`import cv2`
3	`import numpy as np`
4	`from PIL import Image`
5	
6	`lokasifile='Data_Wajah'`
7	`pengenalan = cv2.createLBPHFaceRecognizer()`
8	
9	`def getImagesWithID(lokasifile):`
10	
11	`    wajah=[]`
12	`    idwajah=[]`
13	`    imagePaths=[os.path.join(lokasifile,f) for f`
14	`    in os.listdir(lokasifile)]`
15	
16	`    for imagePath in imagePaths:`
17	`        faceImg =`
18	`        Image.open(imagePath).convert('L');`
19	`        faceNp = np.array(faceImg, 'uint8')`
20	`        ID=int(os.path.split(imagePath)[-`
21	`        1].split('.')[1])`
22	`        wajah.append(faceNp)`
23	`        print ID`

No	*Source Code*
24	idwajah.append(ID)
25	cv2.imshow ("wajah",faceNp)
26	cv2.waitKey(9)
27	return idwajah, wajah
28	
29	idwajah,wajah=getImagesWithID(lokasifile)
30	pengenalan.train(wajah,np.array(idwajah))
31	cv2.imshow ("wajah2",wajah)
32	pengenalan.save('Hasil_Training/LBPH_Wajah.yml')
33	cv2.destroyAllWindows()

Tabel 5.3 akan dijelaskan fungsi setiap kode training data. Proses training dataset ialah proses dimana *dataset* dideskripsiakan menggunakan *Local Binary Patern Histogram*. Proses identifikasi menggunakan *Local Binary Patern* yang nantinya akan mengasilkan keluaran file yang berisikan data hasil identifikasi wajah yang disimpan dalam folder Hasil_Training dengan *format file yml*. File ini yang nantinya akan digunakan untuk standar identifikasi pada proses *recognition* wajah.

Pada program dijelaskan bahwa code akan mengambil data gambar hasil deteksi yang disimpan pada folder Data_Wajah. Proses selanjutnya wajah akan ditraining menggunakan *library* open cv yaitu *Local Binary Patern Histogram* dengan masukan berupa array dari setiap gambar wajah yang berada pada folder Data_Wajah. Hasil perhitungan local binary pattern akan disimpan dalam bentuk histogram dan dalam file yml Hasil_Training. File hasil training selanjutnya akan di *load* pada saat proses *Recognition.* Penulis tidak melakukan penampilan hasil LBP pada program dan penulis juga tidak menampilkan data dalam bentuk histogram, hasil *Local Binary Pattern* hanya disimpan dalam file yml.

5.2.1.3 *Face Recognition*

Tabel 5. 4 Kode Program *Face Recognition*

No	*Source Code*
1	import cv2
2	import numpy as np
3	import pygame
4	import RPi.GPIO as GPIO
5	from time import sleep

No	*Source Code*
6	deteksi=cv2.CascadeClassifier('haarcascade_fronta
7	lface_default.xml');
8	kamera=cv2.VideoCapture(0)
9	pengenalan=cv2.createLBPHFaceRecognizer()
10	pengenalan.load("Hasil_Training\\LBPH_Wajah.yml")
11	id=0
12	while True:
13	ret, gambar =kamera.read()
14	gray=cv2.cvtColor(img,cv2.COLOR_BGR2GRAY)
15	wajah=deteksi.detectMultiScale(gray, 1.3,5)
16	for(x,y,w,h) in faces:
17	id, conf =
18	pengenalan.predict(gray[y:y+h,x:x+w])
19	if(id==1):
20	id = 'Willy'
21	pygame.mixer.music.load("WILLY.wav")
22	pygame.mixer.music.play()
23	sleep (5)
24	pergerakanmotor()
25	elif(id==2):
26	id = 'Silvia'
27	pygame.mixer.music.load("SILVIA.wav")
28	pygame.mixer.music.play()
29	sleep (5)
30	pergerakanmotor()
31	elif(id==3):
32	id = 'Enny'
33	pygame.mixer.music.load("ENNY.wav")
34	pygame.mixer.music.play()
35	sleep (5)
36	pergerakanmotor()
37	elif(id==4):

No	*Source Code*
38	id = 'Fajar'
39	pygame.mixer.music.load("FAJAR.wav")
40	pygame.mixer.music.play()
41	sleep (5)
42	pergerakanmotor()
43	else :
44	pygame.mixer.music.load("ERROR.wav")
45	pygame.mixer.music.play()
46	sleep (5)
47	
48	kamera.release()
49	cv2.destroyAllWindows()

Table 5.4 merupakan kode program untuk *face recognition*. *Face Recognition* yang diterapkan merupakan proses pengenalan wajah menggunakan file hasil identifikasi wajah yang telah disimpan sebelumnya dalam file Data_Wajah yml. Sebenarnya prosesnya hampir sama dengan proses pembuatan dataset. Namun aksi yang diberikan bukan berupa keluaran gambar seperti pada pembuatan data set. Dengan menggunakan data hasil identifikasi dari training dataset, Setiap wajah yang terdeteksi akan langsung dibandingkan dengan cara mencari nilai LBP kemudian dibandingkan dengan nilai LBP yang ada pada file yml. Kemudian dilakukan pemberian identitas berupa siapa pemilik wajah tersebut. Proses ini dilakukan dengan *webcam* berada pada kondisi *standby* dan akan melakukan aksi ketika wajah teridentifikasi kemudian akan kembali pada kondisi stand by. Hasilnya wajah dapat dikenali sesuai identitas yang telah dibuat sebelumnya dan sistem dapat melakukan proses selanjutnya yaitu berupa aksi penyebutan suara, wajah siapa yang ada di depan kamera atau proses notifikasi hasil pengenalan wajah. Kode program akan melakukan semua proses dengan cepat sehingga tidak terasa tiap proses yang terjadi dalam program. Secara teori pengenalan wajah dilakukan dengan mengurangkan 2 buah data yang mirip dan dicari selisih terdekat setelah itu diambil keputusan berupa siapa yang dikenali dalam gambar.

5.2.1.4 Motor Kontroller

Tabel 5. 5 Kode Program Motor Controller

No	*Source Code*
1	GPIO.setmode(GPIO.BOARD)
2	Motor1A = 16
3	

No	*Source Code*
4	`Motor1B = 18`
5	
6	`TriggerM1 = 12`
7	
8	`Motor23A = 38`
9	`Motor23B = 40`
10	
11	`TriggerM23 = 36`
12	
13	`GPIO.setup(Motor1A,GPIO.OUT)`
14	`GPIO.setup(Motor1B,GPIO.OUT)`
15	`GPIO.setup(TriggerM1,GPIO.OUT)`
16	
17	`GPIO.setup(Motor23A,GPIO.OUT)`
18	`GPIO.setup(Motor23B,GPIO.OUT)`
19	`GPIO.setup(TriggerM23,GPIO.OUT)`
20	
21	`def motor():`
22	`    print "buka"`
23	
24	`    GPIO.output(Motor1A,GPIO.HIGH)`
25	`    GPIO.output(Motor1B,GPIO.LOW)`
26	
27	`    GPIO.output(TriggerM1,GPIO.HIGH)`
28	
29	`    sleep(1)`
30	
31	`    GPIO.output(Motor1A,GPIO.LOW)`
32	`    GPIO.output(Motor1B,GPIO.LOW)`
33	
34	`    GPIO.output(TriggerM1,GPIO.HIGH)`
35	
36	`    sleep(1)`
37	
38	`    GPIO.output(Motor23A,GPIO.HIGH)`
39	`    GPIO.output(Motor23B,GPIO.LOW)`
40	
41	`    GPIO.output(TriggerM23,GPIO.HIGH)`
42	
43	`    sleep(5)`
44	
45	`    GPIO.output(Motor23A,GPIO.LOW)`
46	`    GPIO.output(Motor23B,GPIO.LOW)`
47	
48	`    GPIO.output(TriggerM23,GPIO.HIGH)`
49	

No	*Source Code*
50	`    sleep(10)`
51	
52	`    print "tutup"`
53	
54	`    GPIO.output(Motor23A,GPIO.LOW)`
55	`    GPIO.output(Motor23B,GPIO.HIGH)`
56	
57	`    GPIO.output(TriggerM23,GPIO.HIGH)`
58	
59	`    sleep(5)`
60	
61	`    GPIO.output(Motor23A,GPIO.LOW)`
62	`    GPIO.output(Motor23B,GPIO.LOW)`
63	
64	`    GPIO.output(TriggerM23,GPIO.HIGH)`
65	
66	`    sleep(1)`
67	
68	`    GPIO.output(Motor1A,GPIO.LOW)`
69	`    GPIO.output(Motor1B,GPIO.HIGH)`
70	
71	`    GPIO.output(TriggerM1,GPIO.HIGH)`
72	
73	`    sleep(1)`
74	
75	`    print "selesai"`
76	`    GPIO.output(TriggerM1,GPIO.LOW)`
77	`    GPIO.output(TriggerM23,GPIO.LOW)`
78	
79	`    GPIO.cleanup()`

Tabel 5.5 merupakan code untuk mengontrol pemutaran motor dilakukan menggunakan inisialisasi Raspberry pi GPIO. Motor disini bertugas untuk membuka pintu dan menutup pintu dan juga membuka kunci dan menutup kunci. Proses ini dilakukan jika pemberitahuan melalui media suara telah selesai dilakukan. Dan ini merupakan proses terakhir dari rangkaian proses sebelum kembali ke kondisi standby.

Pada kode program yang pertama dilakukan ialah inisialisasi variable menggunakan pin GPIO yang terdapat pada raspberry pi. Pin yang digunakan ialah 12,16,18,36,38,40 hal ini dapat dilihat pada tabel 5.5 kode program baris yang pertama hingga baris ke-11. Kemudian selanjutnya ialah mensetup semua pin sebagai keluaran raspberry pi 3 yang mana keluaran akan bernilai 5 v saat *high* dan bernilai 0 v saat *low.*

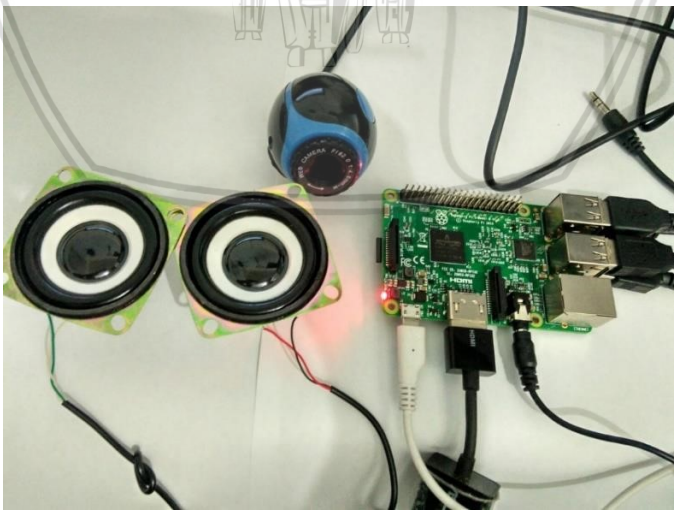
Pada saat membuka pintu motor akan berputar sesuai kondisi, dan urutan kondisinya ialah motor 1 akan berputar selama 1 detik searah jarum jam untuk membuka kunci agar pintu dapat bergeser. Selanjutnya akan ada jeda selama 1 detik sebelum menggerakan motor 2 dan 3, jeda memanfaatkan kondisi dimana kedua output berupa tegangan *low* dan *enable* di *set high*. Kemudian motor 2 dan 3 digerakan salama 5 detik searah jarum jam secara bersamaan untuk menggeser pintu. Akan diberikan jeda lagi selama 10 detik hal ini bertujuan memberi waktu masuk rumah kepada individu yang terdeteksi. Proses pembukaan kunci dan pembukaan pintu dijelaskan pada kode program baris ke-22 hingga baris ke-50.

Penutupan pintu akan berlangsung setelah waktu jeda 10 detik selesai. Dengan urutan proses yaitu menggerakan motor 2 selama 5 detik melawan arah jarum jam untuk menutup pintu. Selanjutnya akan ada jeda selama 1 detik untuk proses selanjutnya yaitu memutar motor 1 melawan arah jarum jam untuk mengunci pintu. Dan pintu akan tertutup sempurna hingga sistem mengenali wajah yang ada di depan kamera. Proses Penutupan pintu dan penguncian kembali dijelaskan pada kode program baris ke-52 hingga baris ke-77. Semua proses pergerakan motor diletakan pada sebuah fungsi yang hanya akan dipanggil ketika sistem sudah mengenali wajah. Fungsi program tidak akan dijalankan jika wajah tidak dikenali sebagai penghuni rumah.

5.2.2 Implementasi Perangkat Keras

Implementasi perangkat keras diberikan dengan cara melampirkan bukti bahwa sistem telah berhasil dibuat berupa gambar ataupun media lainnya yang dapat membuktikan bahwa sistem telah selesai di implementasikan.

5.2.2.1 Implementasi Rangkaian *Webcam*, Raspberry pi dan *Speaker*

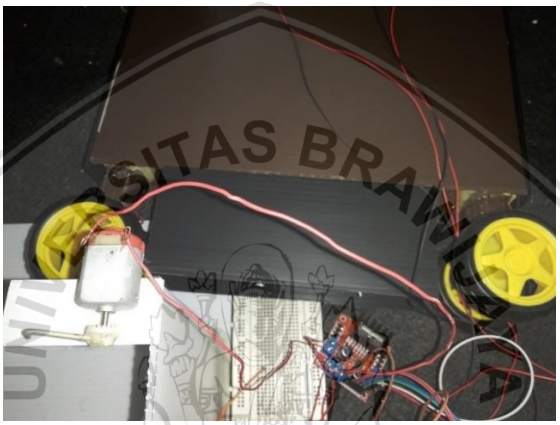


Gambar 5. 27 Foto Rangkaian *Webcam*, Raspberry pi dan *Speaker*

Gambar 5.27 merupakan implementasi rangkaian *Webcam*, Raspbery pi dan *Speaker* dilakukan menggunakan beberapa alat yang siap pakai tanpa harus membuatnya dari awal. Raspberry pi 3 memiliki 4 buah *port USB* dan juga 1 buah

port jack audio 3.5. Dengan memanfaatkan *port* tersebut dirasa sangat efektif karena penulis tidak perlu membuat rangkaian *speaker* dan juga rangkaian *webcam*. *Webcam* yang digunakan ialah *webcam* dengan resolusi kamera 2 *megapixel*. Komunikasi dengan raspberry menggunakan USB untuk komunikasi data dan power *webcam*. *Speaker* pada *webcam* tidak digunakan jadi penulis memutuskan untuk membuang kabel *speaker* pada *webcam.* Speaker yang digunakan ialah *speaker merk* advance dengan dua buah *speaker* yang berfungsi sebagai suara *left* and *right*. Komunikasi speaker menggunakan *jack audio* 3.5 dengan *power supply* menggunakan *usb port*. Baik *Speaker* dan *Webcam* tidak menggunakan komunikasi pin pada raspberry pi 3.

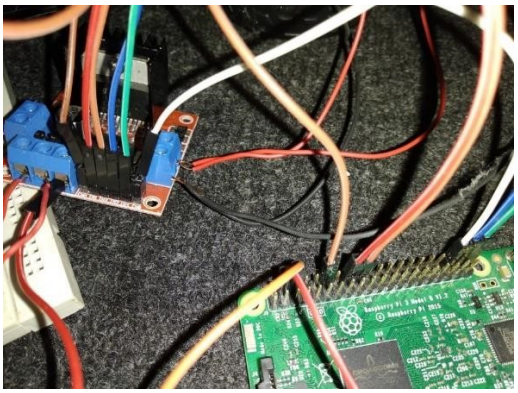
5.2.2.2 Implementasi Rangkaian Motor DC dengan IC L293D



Gambar 5. 28 Foto Rangkaian Motor DC dengan IC L293D

Gambar 5.28 ialah implementasi motor dc dengan IC l293D memanfaatkan media project board dan kabel *jumper*. Peletakan rangkaian nantinya akan berada pada bagian dalam pintu. Kabel *jumper* yang digunakan ialah kabel jumper male to male untuk menghubungkan seluruh motor dengan driver motor. Pada motor 1 kabel yang digunakan ialah sepanjang 30 cm. Sedangkan untuk motor 2 dan 3 dimana telah diletakan pada dudukan motor untuk ban *line tracer* menggunakan kabel jumper sepanjang 70 cm.

5.2.2.3 Implementasi Rangkaian Raspberry pi dengan IC L293D



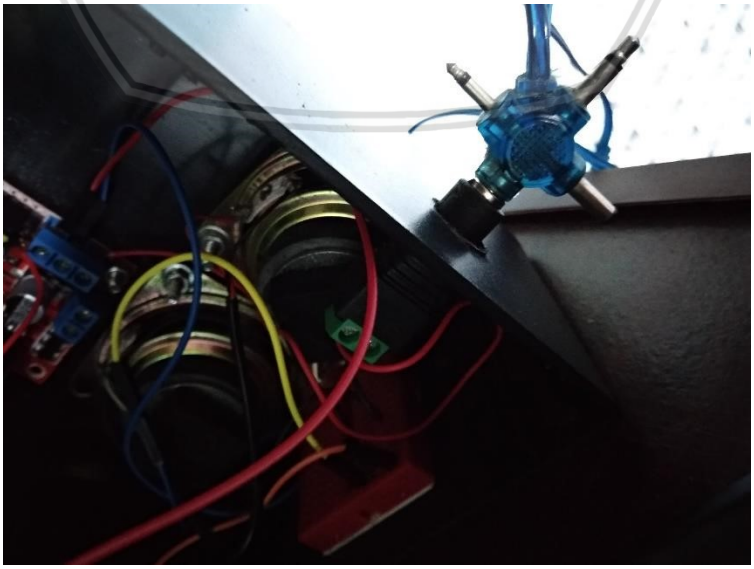
Gambar 5. 29 Foto Raspberry pi dengan IC L293D

Gambar 5.29 merupakan implementasi Raspberry pi dengan IC L293D menggunakan kabel jumper *female to female* untuk komunikasi data. Sedangkan untuk *ground* dan *vcc* menggunakan kabel *jumper female to male*. Penggunaan kabel ini untuk mempermudah pergantian pin jika sewaktu-waktu ada perubahan. Pada tabel 5.6 berikut, merupakan hubungan antara pin pada raspberry pi 3 dengan IC L293D.

Tabel 5. 6 Tabel Komunikasi Pin Raspberry pi dengan IC L293D

Motor	**Pin IC L293D**	**Pin Raspberry pi 3**	**Media**
Motor 1	1	12	*Jumper female to female*
	2	16	*Jumper female to female*
	7	18	*Jumper female to female*
Motor 2	9	36	*Jumper female to female*
	10	38	*Jumper female to female*
	14	40	*Jumper female to female*
Motor 3	9	36	*Jumper female to female*
	10	38	*Jumper female to female*
	14	40	*Jumper female to female*
Vcc 5+	16	4	*Jumper female to male*
Ground	4,5,12,13	6	*Jumper female to male*

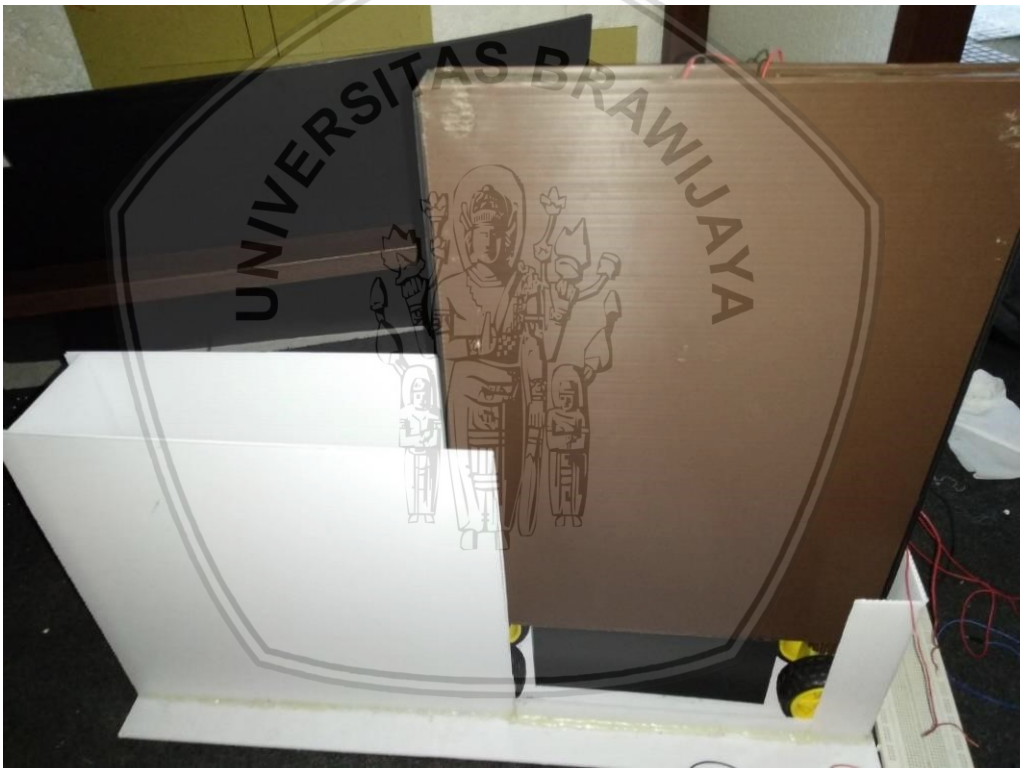
5.2.2.4 Implementasi *Power Supply* Raspberry pi, IC dan Motor DC



Gambar 5. 30 *Power Supply* Raspberry pi, IC dan Motor DC

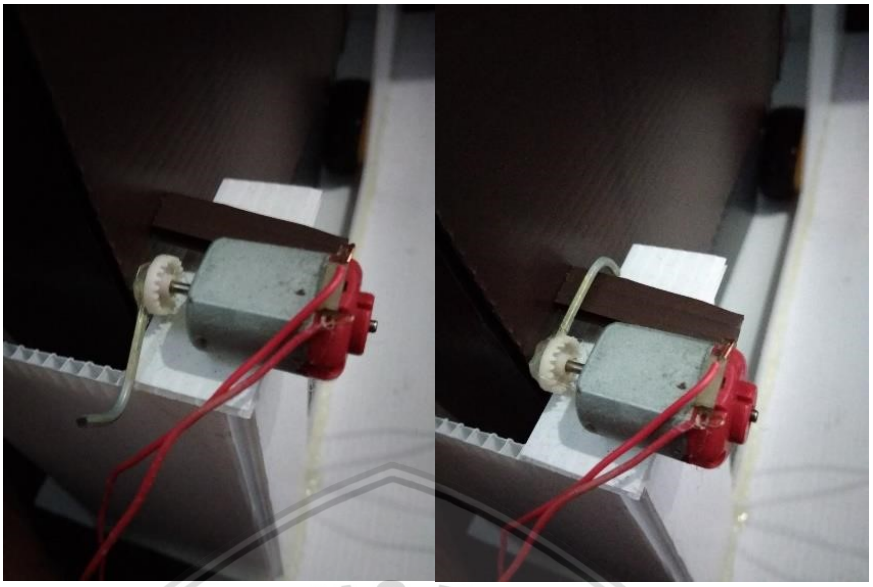
Gambar 5.30 merupakan rangkaian power supply. Untuk *Power supply*, akan ada 2 buah *power* yang dibutuhkan sistem. Yang pertama ialah *Power* untuk raspberry pi 3 sebesar 5.1 *volt* dan 2.5 *ampere*. *Power* untuk raspberry pi menggunakan *power official* bawaan yang mendukung kinerja raspberry pi 3 secara *powerfull* dan stabil. Kemudian *power supply* yang selanjutnya ialah *power supply* untuk icl293d yang nantinya akan digunakan untuk motor. Pin yang digunakan untuk menghubungkan *vcc ground* ialah pin nomor 8 sedangkan pin untuk *ground* ialah pin nomor 4,5,12,13. Untuk mempermudah peletakan *jumper* semua *ground* pada ic L293D dihubungkan menggunakan jumper pada satu baris *project board*. *Power supply* berupa adapter 12 volt, kutub positif akan dihubungkan pada pin no 8 dan kutub negative akan dihubungkan dengan baris *ground* yang lainnya.

5.2.2.5 Implementasi Prototipe Miniatur Pintu



Gambar 5. 31 Foto Pintu Tampak Depan

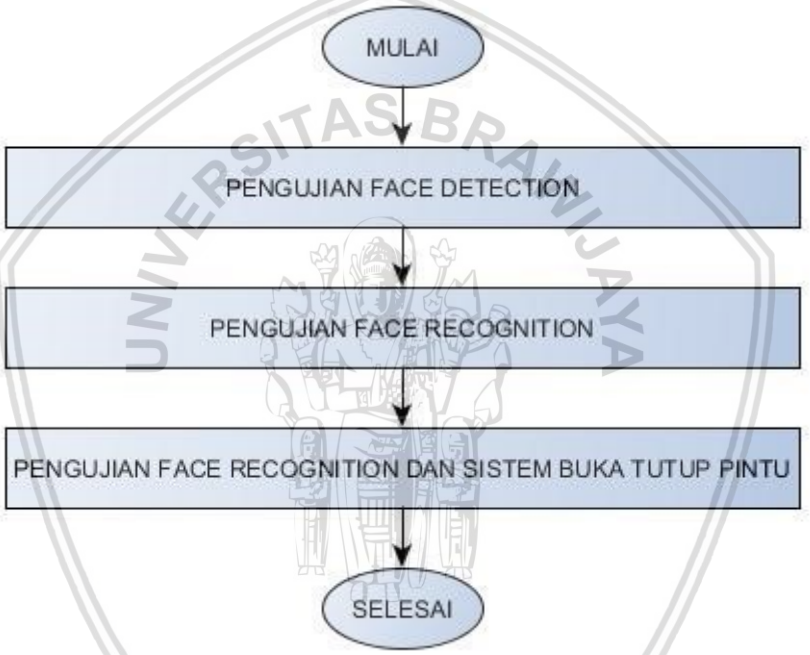
Gambar 5.31 ialah prototipe yang sudah selesai dibuat menggunakan kertas PVC. Kertas PVC digunakan sebagai alas dari prototipe dan juga penyangga pintu gesar agar dapat berdiri tegak. Kertas PVC juga digunakan untuk pembuatan bagian pintu. Semua kabel motor akan disembunyikan pada bagian dalam pintu dan dikeluarkan pada bagian atas pintu agar dapat dengan mudah dihubungkan dengan IC. Dan juga pada motor 1 dimana bertugas mengunci pintu akan diberikan sebuah pengait untuk membuka dan menutup pintu, Gambar 5.32 adalah foto dari pengait pintu yang telah dibuat.



Gambar 5. 32 Pengait ketika dalam posisi terbuka dan terkunci

BAB VI PENGUJIAN DAN ANALISIS

Pengujian dan analisis ialah bab dimana sistem diuji apakah sudah sesuai dengan perancangan dan implementasi sistem. Semua fitur akan diuji apakah sudah bekerja seperti seharusnya. Akan ada 3 pengujian yang akan diujikan kepada sistem, dan ke empat pengujian sudah memenuhi semua fitur yang ada pada sistem. Sistem akan diuji secara berurutan sesuai fungsi sistem dimulai dari pengujian *face detection*, pengujian *face recognition* dan pengujian *face recognition* serta sistem buka tutup pintu. Berikut merupakan urutan pengujian sistem mulai dari awal hingga akhir :



Gambar 6. 1 Alur Pengujian

6.1 Pengujian Akurasi *Face Detection*

6.1.1 Tujuan Pengujian

Tujuan dilakukannya pengujian ini ialah untuk mengetahui fungsi sistem yang pertama yaitu sistem dapat mendeteksi wajah dari sebuah masukan citra. Fungsi *face detection* ialah salah satu fungsi utama agar sistem dapat membuat dataset dan menyimpan wajah dari setiap gambar yang menjadi masukan yang selanjutnya akan dijadikan sebagai pembanding dalam proses pengenalan wajah atau *face recognition.* Tujuan lainnya ialah untuk mengetahui seberapa besar akurasi metode *haar cascade* dalam mendeteksi wajah dari sebuah citra yang akan diambil menggunakan *webcam*/kamera sistem.

6.1.2 Prosedur Pengujian

1. Menyiapkan *code face detection* untuk membuat *data set* sistem dan menyimpannya dalam sebuah folder tertentu.
2. Menyiapkan 4 individu yang akan dijadikan objek pengujian pendeteksian wajah/ *face detection*.
3. Pengambilan wajah/pendeteksian wajah dilakukan pada sebuah ruangan dengan pencahayaan yang cukup.
4. Setiap individu akan melakukan 10 kali pendeteksian dan 1 kali pendeteksian akan diambil 10 sampel gambar.
5. Setiap individu yang akan melakukan pendeteksian wajah berada 1-2 meter didepan kamera.
6. Hasil dari pengambilan gambar akan disimpan pada sebuah folder khusus dan nantinya akan di analisa apakah ada kesalaahan atau error ketika sistem melakukan proses deteksi.
7. Ketika dilakukan pendeteksian, individu yang akan dideteksi melakukan sedikit gerakan menjauhi kamera.
8. Untuk mengetahui tingkat error, penulis akan menyajikan seluruh gambar yang dideteksi sebagai wajah.
9. Sistem dikategorikan sebagai error jika gambar yang terdeteksi ialah bukan wajah.
10. Untuk mengetahui error per individu pada pengujian *face detection* akan digunakan persamaan 6.1 berikut :

$$error\ individu = \frac{total\ gambar\ error}{total\ gambar\ keseluruhan\ individu} \times 100\% \quad \textbf{(6.1)}$$

11. Untuk mengetahui error keseluruhan pada pengujian *face detection* akan digunakan persamaan 6.2 berikut :

$$total\ error = \frac{SUM(error\ individu)}{total\ individu} \quad \textbf{(6.2)}$$

12. Melakukan analisa hasil dan membuat kesimpulan sesuai dengan hasil pengujian.

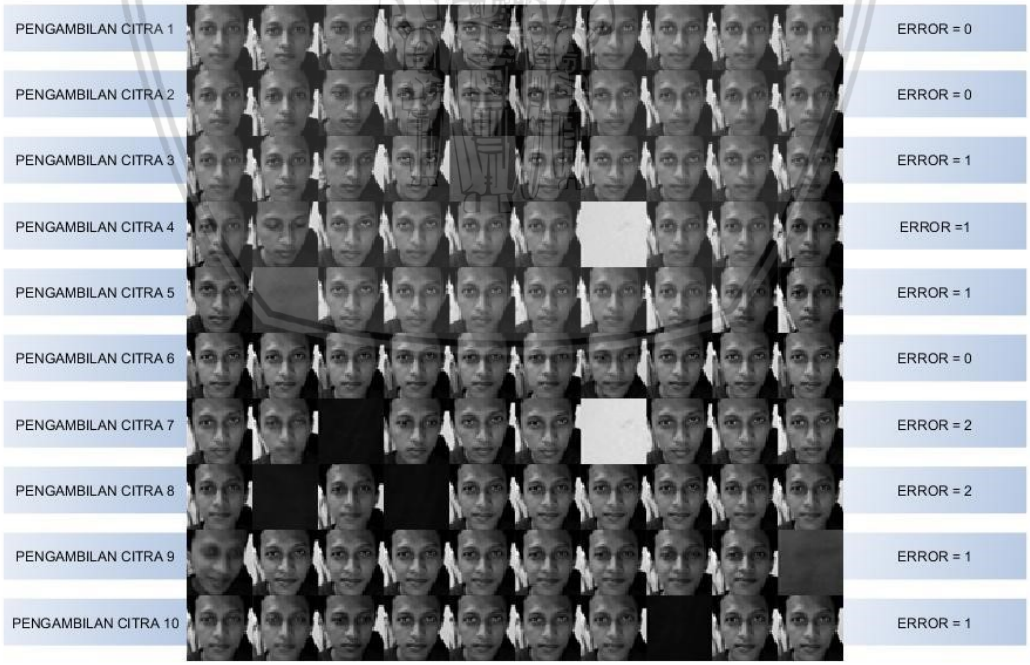
6.1.3 Hasil dan Analisis Pengujian

Setelah dilakukan pengujian terhadap sistem menggunakan 4 individu yang berbeda, Akan dilakukan penggabungan gambar masing-masing individu agar lebih mudah untuk di analisa.

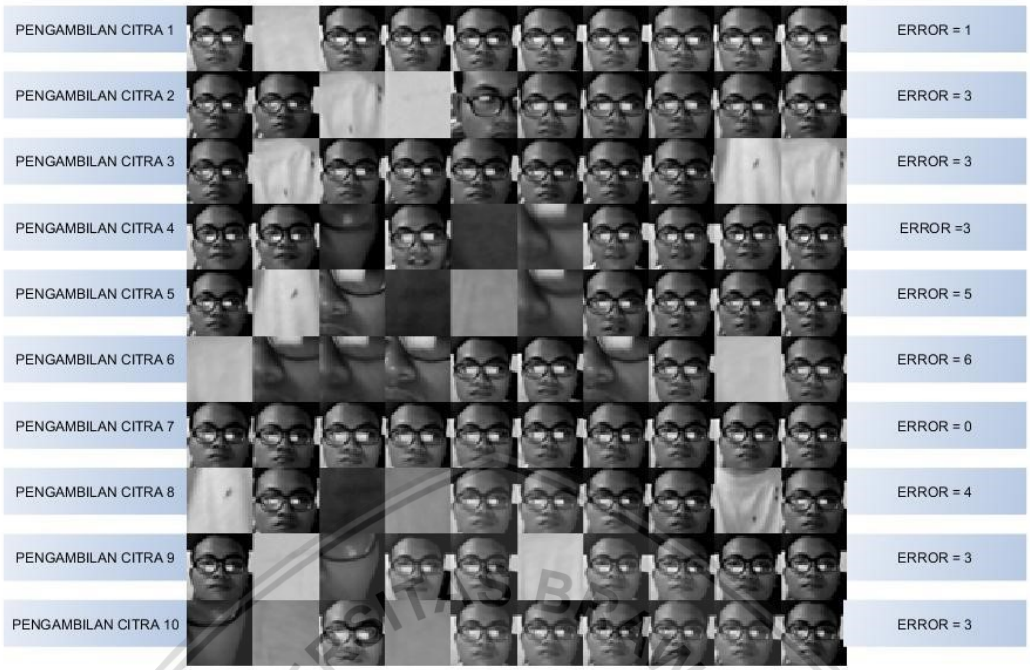
6.1.3.1 Hasil Pengujian Akurasi *Face Detection*

Gambar 6.2 hingga Gambar 6.5 merupakan hasil beserta keterangan jumlah eror setiap individu yang telah melakukan pengujian *face detection*.

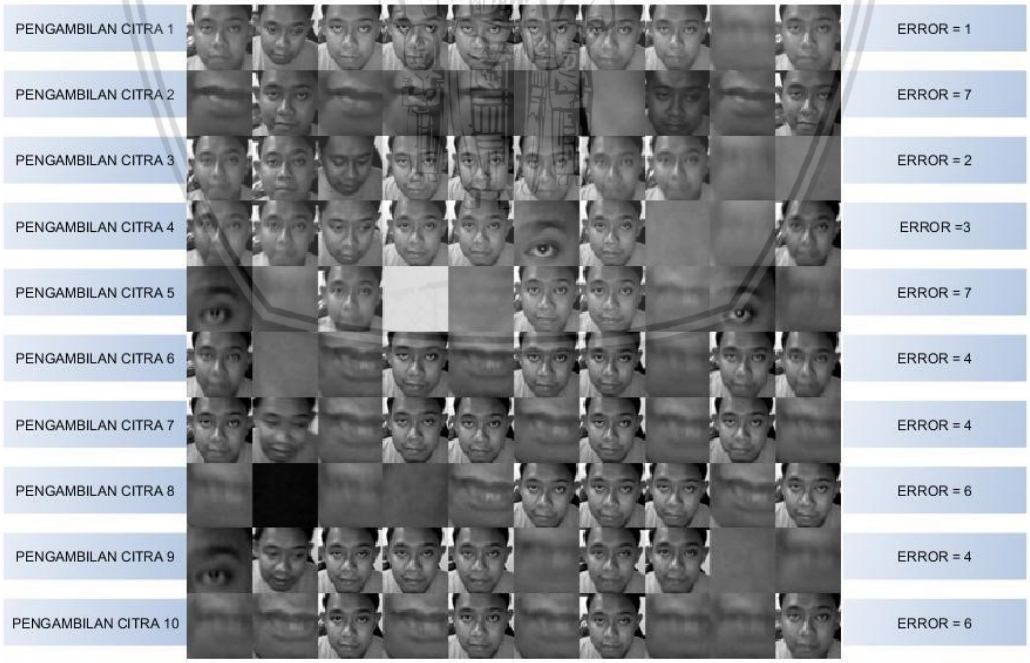
Gambar 6. 2 Hasil Deteksi Individu 1



Gambar 6. 3 Hasil Deteksi Individu 2



Gambar 6. 4 Hasil Deteksi Individu 3



Gambar 6. 5 Hasil Deteksi Individu 4

6.1.3.2 Analisa Pengujian Buka Tutup Pintu

Dari hasil pengujian di atas, berikut merupakan Analisa hasil pengujian *face detection* yang telah dilakukan. Dapat dilihat pada hasil pengujian bahwa setiap individu yang melakukan pengujian memiliki jumlah eror yang berbeda-beda. Hal ini dikarenakan setiap individu memiliki wajah yang berbeda dan juga melakukan gerakan menjauhi kamera yang berbeda-beda. Untuk mengetahui persentase eror masing-masing individu dilakukan perhitungan menggunakan persamaan 6.1 sebagai berikut.

$$error\ individu = \frac{total\ gambar\ error}{total\ gambar\ keseluruhan\ individu} \times 100\%$$

Dibutuhkan beberapa nilai untuk mengisi variabel diatas, Dari hasil pengujian berikut merupakan data hasil uji masing-masing individu.

a. Individu 1
   Total gambar error : 11
   Total seluruh gambar : 100
b. Individu 2
   Total gambar error : 9
   Total seluruh gambar : 100
c. Individu 3
   Total gambar error : 31
   Total seluruh gambar : 100
d. Individu 4
   Total gambar error : 44
   Total seluruh gambar : 100

Dari data yang telah didapat yang maka error masing-masing individu ialah sebagai berikut :

a. Individu 1

$$error\ individu = \frac{total\ gambar\ error}{total\ gambar\ keseluruhan\ individu} \times 100\%$$

$$error\ individu = \frac{11}{100} \times 100\%$$

$$error\ individu = 11\%$$

b. Individu 2

$$error\ individu = \frac{total\ gambar\ error}{total\ gambar\ keseluruhan\ individu} \times 100\%$$

$$error\ individu = \frac{9}{100} \times 100\%$$

$$error\ individu = 9\%$$

c. Individu 3

$$error\ individu = \frac{total\ gambar\ error}{total\ gambar\ keseluruhan\ individu} \times 100\%$$

$$error\ individu = \frac{31}{100} \times 100\%$$

$$error\ individu = 31\%$$

d. Individu 4

$$error\ individu = \frac{total\ gambar\ error}{total\ gambar\ keseluruhan\ individu} \times 100\%$$

$$error\ individu = \frac{44}{100} \times 100\%$$

$$error\ individu = 44\%$$

Setelah mengetahui error masing-masing individu berikutnya ialah mengetahui error keseluruhan *face detection*. Persamaan 6.2 merupakan rumus untuk mengetahui tingkat error keseluruhan *face detection*.

$$total\ error = \frac{SUM(error\ individu)}{total\ individu}$$

Menggunakan nilai yang telah didapat sebelumnya maka total error yang terjadi pada pengujian *face detection* ialah sebagai berikut :

$$total\ error = \frac{SUM(error\ individu)}{total\ individu}$$

$$total\ error = \frac{9\% + 11\% + 31\% + 44\%}{4}$$

$$total\ error = \frac{95\%}{4}$$

$$total\ error = 23.75\%$$

Didapat angka 23.75% total error, Jadi hasil pengujian akurasi *face detection* ialah sebesar 76.25%. Angka tersebut didapat dari 100% dikurangi dengan total error. Nilai akurasi yang didapat masih cukup aman karena angka masih diatas 60%. Sebagian besar error yang terjadi dikarenakan posisi pengambilan gambar dan juga gerakan yang dilakukan individu ketika *face detection* dilakukan. Beberapa gambar yang dideteksi sebagai wajah antara lain ialah mata, baju yang dikenakan, leher dan beberapa bagian tubuh lainnya.

6.2 Pengujian Akurasi *Face Recognition*

6.2.1 Tujuan Pengujian

Pengujian akurasi *face recognition* dilakukan untuk mengetahui persentase besaran seberapa kuat sistem bisa mengenali wajah dan juga untuk mengetahui apakah ada kesalahan pada sistem dimana wajah selain dari yang ada pada datasheet dapat dikenali. Pengujian ini sangatlah penting karena gagal atau tidaknya kinerja sistem, akan tergantung dengan tingkat akurasi pengenalan wajah.

6.2.2 Prosedur Pengujian

1. Menyiapkan 4 orang yang akan bertindak sebagai penghuni rumah yang terdiri dari dua orang perempuan dan dua orang laki-laki. Mencatat setiap identitas orang yang telah dipilih dan mengelompokannnya ke dalam anggota penghuni rumah.
2. Melakukan pengambilan gambar wajah pada 4 orang yang telah dipilih sebangak 10 kali pada masing-masing orang dengan pandangan wajah yang berbeda-beda sesuai dengan perancangan sistem. Prosedur pengambilan gambar untuk *dataset* ialah individu berada 50 cm di depan kamera.
3. Memberikan identitas pada setiap gambar sesuai nama yang telah di catat sebelumnya.
4. Melakukan penyimpanan gambar yang telah di ambil pada folder menggunakan *code* yang telah disiapkan.
5. Menyiapkan 16 orang yang akan bertindak sebagai orang asing atau bukan penghuni rumah.
6. Mencatat setiap identitas berupa nama pada setiap orang yang telah dipilih sebagai bukan penghuni.
7. Melakukan pengujian akurasi dengan cara setiap penghuni rumah melakukan pengenalan wajah. Jika dikenali berarti sistem berjalan dengan benar. Namun jika tidak dikenali atau dikenali sebagai orang lain maka sistem melakukan kesalahan.
8. Melakukan pencatatan hasil dan akan disajikan dalam sebuah tabel. Penyajian tabel akan terdiri dari 4 kolom berisi No, Nama, Benar dan Salah.
9. Pada kolom No akan diisi angka dari 1 hingga 20. Pada kolom Nama akan diisi Nama setiap orang yang menjadi objek pengujian dimulai dari 4 orang penghuni kemudian diikuti dengan 16 orang yang bukan penghuni.
10. Jika penghuni rumah, akan diberi tanda berupa tulisan “dikenali” pada kolom “benar” jika wajah dikenali. Namun jika tidak dikenali akan diberi tanda berupa “tidak dikenali” pada kolom “salah”.

11. Untuk mengetahui akurasi *face recognition* dilakukan dengan menggunakan persamaan 6.3 berikut:

$$\boldsymbol{X} = \frac{\boldsymbol{Z}}{\boldsymbol{Y}} \times \mathbf{100}\% \quad \textbf{(6.3)}$$

$\boldsymbol{X} =$ Akurasi *Face Recognition*

$\boldsymbol{Y} =$ Jumlah Pengenalan yang Benar

$\boldsymbol{Z} =$ Total Individu

13. Melakukan analisa hasil dan membuat kesimpulan sesuai hasil pengujian.

6.2.3 Data Set Pengujian

Untuk menjalankan pengujian *face recognition* dibutuhkan dataset berupa wajah penghuni rumah sebagai pembanding untuk pengenalan wajah. Berikut merupakan data nama penghuni rumah :

1. Willy Andika Putra

   Dilakukan 10 pengambilan gambar untuk *data set* dengan 9 arah pandang yang berbeda. Gambar 6.2 merupakan *data set* Willy Andika Putra yang akan bertindak sebagai penghuni rumah 1 :

Gambar 6. 6 Data Set Wajah Willy

2. Silvia Devi Enggarwati

   Dilakukan 10 pengambilan gambar untuk *data set* dengan 9 arah pandang yang berbeda. Gambar 6.3 merupakan *data set* Silvia Devi Enggarwati yang akan bertindak sebagai penghuni rumah 2:

Gambar 6. 7 Data Set Wajah Silvia

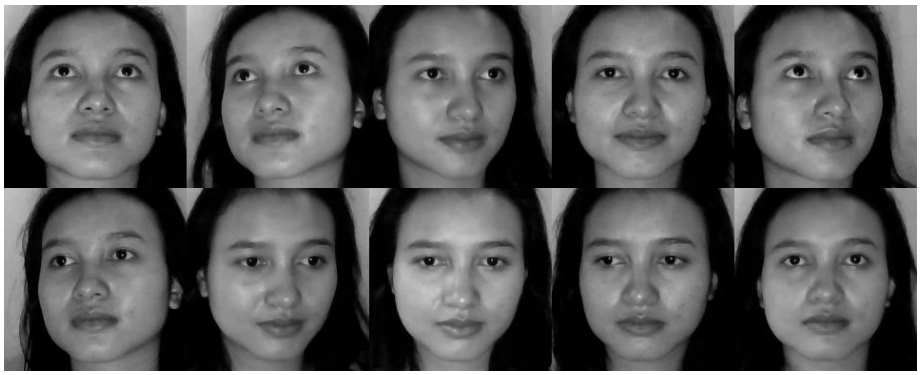
3. Muhammad Fajarudin

   Dilakukan 10 pengambilan gambar untuk *data set* dengan 9 arah pandang yang berbeda. Gambar 6.4 merupakan *data set* Muhammad Fajarudin yang akan bertindak sebagai penghuni rumah 3:

Gambar 6. 8 Data Set Wajah Fajar

4. Enny Trisnawati

   Dilakukan 10 pengambilan gambar untuk *data set* dengan 9 arah pandang yang berbeda. Gambar 6.5 merupakan *data set* Enny Trisnawati yang akan bertindak sebagai penghuni rumah 4:



Gambar 6. 9 Data Set Wajah Enny

Berikut merupakan data nama bukan penghuni rumah:

1. M. Zamroni Amali
2. Adnan Mahfudzon
3. Delta Rudi
4. Billy Gusparentaqi
5. Hariyogi Vernando
6. Bagus Cakra Jati Kesuma
7. Indera Ulung Mahendra
8. Rizky Putra Wijaya
9. M Alfian A. P.
10. Falachudin Akbar
11. Rinaldi Albert
12. Ana Fitriani
13. Ayu Mufid
14. Anggifa
15. Shelsa
16. Nafisa

6.2.4 Hasil dan Analisis Pengujian

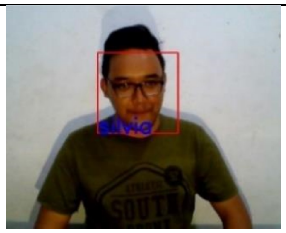
Hasil pengujian akurasi *face recognition* akan disajikan dalam bentuk tabel agar lebih mudah dimengerti oleh pembaca. Tabel penyajian akan memiliki 4 kolom yaitu nomor, nama, status benar dan status salah.

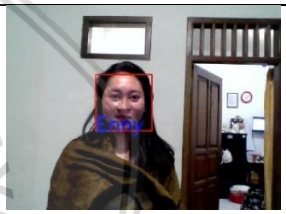
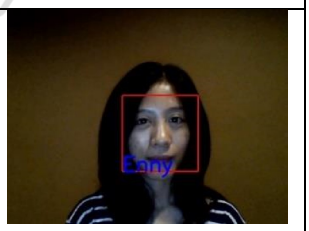
6.2.4.1 Hasil Pengujian Akurasi *Face Recognition*

Tabel 6. 1 Tabel Hasil Pengujian Akurasi *Face Recognition*

No	Nama	Benar	Salah
1	Willy Andika Putra	Sesuai	
			
2	Silvia Devi Enggarwati	Sesuai	
			

3	Muhammad Fajarudin		Tidak Sesuai
			
4	Enny Trisnawati	Sesuai	
			
5	M.Zamroni Amali	Sesuai	
			
6	Adnan Mahfudzon		Tidak Sesuai
			
7	Delta Rudi	Sesuai	
			
8	Billy Gusparentaqi	Sesuai	
			

9	Hariyogi Vernando		Tidak Sesuai
			
10	Bagus Cakra Jati Kesuma		Tidak Sesuai
			
11	Indera Ulung Mahendra	Sesuai	
			
12	Rizky Putra Wijaya	Sesuai	
			
13	M Alfian A. P.	Sesuai	
			
14	Falachudin Akbar	Sesuai	
			

15	Rinaldi Albert	Sesuai	
			
16	Ana Fitriani		Tidak Sesuai
			
17	Ayu Mufid		Tidak Sesuai
			
18	Anggifa	Sesuai	
			
19	Shelsa		Tidak Sesuai
			
20	Nafisa	Sesuai	
			

6.2.4.2 Analisa Pengujian Akurasi *Face Recognition*

Dari hasil pengujian pada tabel 6.1, berikut merupakan Analisa hasil pengujian. Pertama untuk mengetahui tingkat akurasi dari metode yang digunakan penulis menggunakan rumus pada persamaan 6.1.

$$\boldsymbol{X} = \frac{\boldsymbol{Z}}{\boldsymbol{Y}} \times \mathbf{100}\%$$

$\boldsymbol{X}$ = Akurasi *Face Recognition*

$\boldsymbol{Y}$ = Jumlah Pengenalan yang Benar

$\boldsymbol{Z}$ = Total Individu

Pada tabel 6.1 menunjukan bahwa Jumlah Pengenalan yang benar ada sebanyak 13 wajah. Sedangkan untuk pengenalan yang salah ada 7 Wajah. Total individu yang digunakan dalam sample pengujian ada sebanyak 20 orang. Jadi jika merujuk rumus menentukan akurasi pengenalan diatas maka akurasi pengenalan wajah adalah sebagai berikut :

$\boldsymbol{X}$ = Akurasi *Face Recognition* = ?

$\boldsymbol{Y}$ = Jumlah Pengenalan yang Benar = 13

$\boldsymbol{Z}$ = Total Individu = 20

$$\boldsymbol{X} = \frac{\mathbf{13}}{\mathbf{20}} \times \mathbf{100}\%$$

$$\boldsymbol{X} = \mathbf{65}\%$$

Pengujian *face recognition* dilakukan dengan asumsi bahwa pada proses pembuatan data set sistem tidak memiliki error. Kesimpulan dari pengujian kedua ialah, akurasi pengenalan wajah cukup rendah untuk digunakan sebagai sistem keamanan rumah. Hampir setiap individu berkacamata mengalami eror pada saat proses pengenalan. *Metode Haar Cascade* sangat mudah mengenali ketika kondisi cahaya cukup. Error yang terjadi merupakan kesalahan sistem dimana penghuni dikenali sebagai bukan penghuni dan bukan penghuni dikenali sebagai penghuni. Ini cukup krusial jika digunakan sebagai sistem keamanan. Namun angka 65% dapat dibilang cukup untuk sebuah sistem pengenalan dikarenakan masih diatas 60%.

6.3 Pengujian *Face Recognition* dan Sistem Buka Tutup Pintu

6.3.1 Tujuan Pengujian

Tujuan dari pengujian ini ialah untuk mengetahui tingkat akurasi sistem dengan cara membuat sebuah simulasi kasus pengenalan dimana akan dilakukan pada 4 orang penghuni rumah dan 16 orang bukan penghuni rumah. Dan menguji kinerja sistem buka tutup pintu dengan cara apakah pintu terbuka sempurna ketika wajah dikenal atau tidak dan apakah pemberitahuan suara terhadap siapa yang dikenali sudah tepat. Untuk lebih lengkapnya dapat dilihat pada prosedur pengujian.

6.3.2 Prosedur Pengujian

1. Membuat perancangan sistem mulai dari rangkaian *hardware*, peletakan pada prototipe dan juga pemberian *power supply*.
2. Menyiapkan kode program untuk pembuatan *dataset*, training data set, dan *face recognition* program.
3. Menyiapkan 4 buah *file audio* yang berisi informasi identitas pemilik rumah.
4. Menyiapkan 4 individu yang mana dua laki-laki dan dua wanita yang akan berperan sebagai pemilik rumah.
5. Menyiapkan 16 orang individu secara acak untuk mencoba membobol sistem menggunakan wajah mereka.
6. Sistem bekerja dengan benar jika wajah dikenali sebagai pemilik rumah, Menyebutkan siapa yang dikenali, Membuka kunci pintu, Menggeser pintu hingga terbuka sempurna, Memberi jeda 10 detik untuk orang masuk, Menggeser pintu hingga pintu tertutup sempurna, Mengunci pintu dan kembali ke kondisi idle.
7. Sistem bekerja dengan benar jika wajah tidak dikenali sebagai pemilik rumah, Sistem akan menyebutkan maaf wajah anda tidak dikenali, Dan akan kembali ke kondisi idle
8. Hasil uji akan disajikan dalam bentuk table dengan kolom berurutan sesuai dengan kasus diatas.
9. Menentukan persentase error pengenalan wajah menggunakan persamaan 6.4 berikut :

$$P = \frac{Q}{R} \times 100\% \quad \textbf{(6.4)}$$

P = Persentase Error Pengenalan Wajah
Q = Total Pengenalan Error
R = Total Pengujian

10. Menentukan persentase error buka tutup pintu menggunakan persamaan 6.5 hingga 6.7 sebagai berikut :

$$A = \frac{B}{C}\ x\ 100\% \quad \textbf{(6.5)}$$

A = Persentase Error Motor 1
B = Total Error Motor 1
C = Total Pengujian Motor 1

$$D = \frac{E}{F}\ x\ 100\% \quad \textbf{(6.6)}$$

D = Persentase Error Motor 2 dan 3
E = Total Error Motor 2 dan 3
F = Total Pengujian Motor 2 dan 3

$$G = \frac{A+D}{2} \quad \textbf{(6.7)}$$

A = Persentase Error Motor 1
D = Persentase Error Motor 2 dan 3
G = Persentase Error Motor

11. Menentukan total persentase error keseluruhan sistem menggunakan persamaan (6.8) sebagai berikut :

$$H = \frac{P+G}{2} \quad \textbf{(6.8)}$$

H = Persentase Error Keseluruhan Sistem
P = Error pengenalan wajah
G = Persentase Error Motor

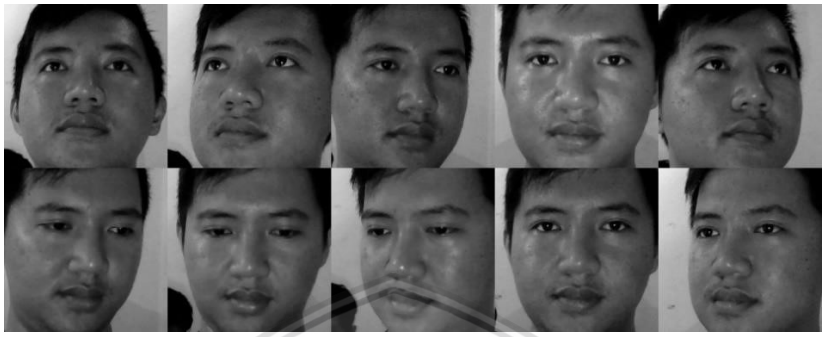
12. Melakukan analisis terhadap hasil pengujian dan membuat kesimpulan dari hasil pengujian.

6.2.3 Data Set Pengujian

Setiap individu melakukan *face recognition* di depan wajah secara bergantian. Ketika sistem mengalami error pada satu proses maka pengujian pada individu akan di stop dan dilanjutkan ke individu selanjutnya. Data yang akan di catat dan dimasukan ke dalam tabel ialah berupa tanda di bagian sistem yang mengalami error. Kolom pada tabel akan sesuai dengan urutan kejadian sistem ketika dijalankan. Untuk individu pengujian, akan menggunakan individu yang sama pada pengujian akurasi *face recognition,* hal ini dilakukan untuk effisiensi waktu agar tidak melakukan pengambilan data set baru. Berikut merupakan data individu yang akan dijadikan sebagai data set :

1. **Willy Andika Putra**

   Dilakukan 10 pengambilan gambar untuk data set dengan 9 arah pandang yang berbeda. Gambar 6.10 merupakan *dataset* Willy Andika Putra yang akan bertindak sebagai penghuni rumah 1:



Gambar 6. 10 Data Wajah Willy

2. **Silvia Devi Enggarwati**

   Dilakukan 10 pengambilan gambar untuk data set dengan 9 arah pandang yang berbeda. Gambar 6.11 merupakan *dataset* Silvia Devi Enggarwati yang akan bertindak sebagai penghuni rumah 2:

Gambar 6. 11 Data Wajah Silvia

3. **Muhammad Fajarudin**

   Dilakukan 10 pengambilan gambar untuk data set dengan 9 arah pandang yang berbeda. Gambar 6.12 merupakan *dataset* Muhammad Fajarudin yang akan bertindak sebagai penghuni rumah 3:

Gambar 6. 12 Data Wajah Fajar

4. Enny Trisnawati

   Dilakukan 10 pengambilan gambar untuk data set dengan 9 arah pandang yang berbeda. Gambar 6.13 merupakan *dataset* Enny Trisnawati yang akan bertindak sebagai penghuni rumah 4:

Gambar 6. 13 Data Wajah Enny

Berikut Merupakan Data Nama Bukan Penghuni Rumah:

1. M. Zamroni Amali
2. Adnan Mahfudzon
3. Delta Rudi
4. Billy Gusparentaqi
5. Hariyogi Vernando
6. Bagus Cakra Jati Kesuma
7. Indera Ulung Mahendra
8. Rizky Putra Wijaya
9. M Alfian A. P.
10. Falachudin Akbar
11. Rinaldi Albert
12. Ana Fitriani
13. Ayu Mufid
14. Anggifa
15. Shelsa
16. Nafisa

6.2.4 Hasil dan Analisis Pengujian

Hasil pengujian keseluruhan sistem akan disajikan dalam bentuk tabel agar lebih mudah dimengerti oleh pembaca. Tabel penyajian akan memiliki 7 kolom yang akan diurutkan sesuai skema pengujian

6.2.4.1 Hasil Pengujian Keseluruhan Sistem

Tabel 6. 2 Tabel Error Keseluruhan Sistem

No	Nama	*Face Recognition*	Motor 1 (Buka Kunci)	Motor 2 dan 3 (Buka Pintu)	Motor 2 dan 3 (Tutup Pintu)	Motor 1 Mengunci
1	Willy Andika Putra	Sesuai	V	V	V	V
2	Silvia Devi Enggarwati	Sesuai	V	V	V	V
3	M. Fajarudin	Tidak sesuai	X	X	X	X
4	Enny Trisnawati	Sesuai	V	V	V	V
5	M.Zamroni Amali	Sesuai	V	V	eror	X
6	Adnan Mahfudzon	Tidak sesuai	X	X	X	X
7	Delta Rudi	Sesuai	V	V	V	V
8	Billy Gusparentaqi	Sesuai	V	V	V	V
9	Hariyogi Vernando	Sesuai	V	V	V	V
10	Bagus Cakra J.K.	Sesuai	V	V	V	V
11	Indera Ulung M.	Tidak sesuai	X	X	X	X
12	Rizky Putra Wijaya	Sesuai	V	V	V	V
13	M Alfian A. P.	Sesuai	V	V	V	V
14	Falachudin Akbar	Tidak sesuai	X	X	X	X
15	Rinaldi Albert	Tidak sesuai	X	X	X	X
16	Ana Fitriani	Tidak sesuai	X	X	X	X
17	Ayu Mufid	Sesuai	V	V	eror	X
18	Anggifa	Sesuai	V	V	V	V
19	Shelsa	Tidak sesuai	X	X	X	X
20	Nafisa	Sesuai	V	eror	X	X

6.2.4.2 Analisa Pengujian Keseluruhan Sistem

Dari hasil pengujian diatas kita dapat melihat bahwa hanya beberapa percobaan yang berhasil tanpa error , dan sebagian besar error terjadi pada saat *face recognition* dilakukan. Motor 1 tidak mengalami error dan ada beberapa eror yang terjadi pada motor 2 berupa tidak sempurna nya pintu ketika terbuka. Berikut merupakan perhitungan total error keseluruhan sistem :

Untuk mengetahui persentase error pengenalan wajah menggunakan rumus sesuai prosedur pengujian sebagai berikut:

a. **Error *Face Recognition***

$$P = \frac{Q}{R}\ x\ 100\%$$

$P =$ Persentase error pengenalan wajah
$Q =$ Total Pengenalan Error
$R =$ Total Pengujian

$P = ?$
$Q = 8$
$R = 20$

$$P = \frac{8}{20}\ x\ 100\%$$

$$P = 40\%$$

Untuk mengetahui persentase motor menggunakan rumus sesuai prosedur pengujian sebagai berikut:

a. Error Motor 1 :

$$A = \frac{B}{C}\ x\ 100\%$$

$A =$ Persentase Error Motor 1
$B =$ Total Error Motor 1
$C =$ Total Pengujian Motor 1

$A = ?$
$B = 0$
$C = 12$

$$A = \frac{0}{12}\ x\ 100\%$$

$$A = 0\%$$

b. Error Motor 2 :

$$D = \frac{E}{F}\ x\ 100\%$$

$D =$ Persentase Error Motor 2
$E =$ Total Error Motor 2
$F =$ Total Pengujian Motor 2

$D = ?$
$E = 3$
$F = 12$

$$D = \frac{3}{12}\ x\ 100\%$$

$$D = 25\%$$

c. Total Error Motor :

$$G = \frac{A + D}{2}$$

$G =$ Persentase Error Motor
$D = 0\%$
$A = 25\%$

$$G = \frac{0\%+25\%}{2}$$

$$G = 12.5\%$$

d. Total Error Keseluruhan Sistem

$$H = \frac{P + G}{2}$$

$H =$ Persentase Error Keseluruhan Sistem
$P = 40\%$
$G = 12.5\%$

$$H = \frac{40\% + 12.5\%}{2}$$

$$H = 26.25\%$$

Dari perhitungan diatas dapat kita lihat bahwa Total error keseluruhan sistem ialah sebesar 26.25%. Kesimpulannya, sistem dapat dibilang cukup untuk sebuah sistem keamanan. Karena tingkat error yang dimiliki ialah 26.25% dan ini masih dibawah 50%. Sebagian besar error terjadi pada saat proses pengenalan wajah. Hal ini dikarenakan metode yang digunakan sangat rentan terhadap noise. Untuk error pada motor, hanya terjadi sedikit dan itupun karena faktor-faktor non teknis seperti pintu yang nyangkut ketika digerakan oleh motor dan juga pintu yang keluar dari rel yang telah disediakan.

BAB VII PENUTUP

7.1 Kesimpulan

1. Cara mengimplementasikan *face recognition* ke dalam raspberry pi yang paling mudah dan efisien ialah menggunakan *Library Open CV* yang tersedia bebas dan dapat digunakan secara gratis. Penggunaan *Library Open CV* harus dilakukan menggunakan Bahasa yang *compatible* dengan Raspbery pi dan juga *library* itu sendiri. Bahasa pemrograman python merupakan Bahasa pemrograman yang dapat digunakan karena Bahasa pemrograman python sudah tersedia di dalam raspberry pi dan juga *Library Haar Cascade Classifier* tersedia untuk Bahasa pemrograman python.
2. Akurasi pendeteksian wajah menggunakan metode *haar-cascade* ialah sebesar 76.25%. Sebagian besar kekeliruan atau error yang terjadi ialah dikarenakan perubahan posisi atau pergerakan seseorang yang akan dideteksi. Hasil pengujian menyatakan sebagian besar yang keliru dideteksi ialah bagian mata, bibir dan baju yang dikenakan oleh individu yang akan dideteksi.
3. Akurasi pengenalan wajah atau *face recognition* menggunakan metode *haar cascade classifier* dan *local binary pattern* ialah sebesar 65%. Error yang terjadi ialah mengenali yang bukan penghuni sebagai penghuni dan juga sebaliknya mengenali penghuni sebagai bukan penghuni. Biasanya hal ini dikarenakan faktor pencahayaan yang kurang tepat pada saat proses pengenalan wajah atau *face recognition*.
4. Cara menghubungkan *face recognition* dengan sistem buka tutup pintu rumah ialah dengan menggunakan motor DC yang akan dijadikan sebagai roda pintu pada prototipe. Motor DC digerakan menggunakan PIN GPIO yang ada pada Raspberry pi. Bahasa pemrograman yang digunakan untuk menggerakan motor ialah Bahasa pemrograman Python. Hal ini dilakukan agar sistem buka tutup pintu dapat dihubungkan dengan *face recognition* yang mana merupakan inti dari penelitian ini.

7.2 Saran

Untuk Pengembangan lebih lanjut, penulis memiliki beberapa saran pengembangan berdasarkan kendala saat pengerjaan maupun hasil pengujian. Berikut merupakan beberapa saran dari penulis untuk para pengembang tekhnologi :

1. Penggunaaan metode yang memiliki tingkat akurasi yang lebih tinggi bisa membuat sistem lebih baik.
2. Penambahan pada sisi alarm dirasa dapat membuat tingkat keamanan yang dimiliki sistem akan lebih baik dari sekarang.
3. Penambahan Pemberitahuan berupa GSM modul yang dapat memberitahu penghuni secara langsung dirasa dapat menambah tingkat keamanan dari sistem ini.

DAFTAR PUSTAKA

Agung, H., 2016. *Ilmusahid.* [Online] Available at: http://www.ilmusahid.com/2016/08/pengertian-webcam-fungsi-webcam dan. html [Accessed 20 April 2018].

Ahonen, T., Hadid, A. & Pietik¨aine, M., 2004. *Face Recognition With Local Binary Patterns,* Oulu: Machine Vision Group Infotech Oulu.

Ali, 2017. *Anriz Global Connection.* [Online] Available at: https://komputerunbk.com/blog/pengertian-dan-fungsi-webcam-dalam-komputer/ [Accessed 28 April 2018].

Anon., 2012. *Elektronika Dasar.* [Online] Available at: http://elektronika-dasar.web.id/driver-motor-dc-l293d/ [Accessed 21 April 2018].

Arief, 2012. *2018 Informatika.* [Online] Available at: http://informatika.web.id/category/citra-digital/page/5 [Accessed 6 April 2018].

Biglari, M., Mirzaei, F. & Ghavidel N, J., 2014. Persian/Arabic Handwritten Digit Recognition Using Local Binary Pattern. *International Journal of Digital Information and Wireless Communication,* IV(4), pp. 486-492.

Chan, P., 2015. *Element 14.* [Online] Available at: https://www.element14.com/community/docs/DOC-73950/l/raspberry-pi-3-model-b-gpio-40-pin-block-pinout [Accessed 6 April 2018].

Derpanis, K. G., Leung, E. T. & Sizintev, M., 2007. *Fast Scale-Space Feature Representations by Generalized Integral,* Toronto: Center Of Vision Research Yosk University.

Fauzi, R. R., Siregar, S. & Soegiarto, D., 2011. *Sistem Pengendali Robot Mobil Berbasis Mikrokontroller ATMEGA16 Dengan Antarmuka RJ45,* Bandung: Politeknik Telkom Bandung Library.

Ferreira, A., 2007. *A Survey on Boosting Algorithms for Supervised Learning,* s.l.: s.n.

Ghimire, R., 2017. *The Debuggers.* [Online] Available at: http://thedebuggers.com/easy-face-detection-using-jquery/ [Accessed 7 April 2018].

Hendry, J., 2012. *Integral Image (Viola Jones) Theory And Matlab.* Yogyakarta, EE & IT UGM.

Hjelm˚as, E. & Low, B. K., 2001. Face Detection: A Survey. *Computer Vision And Image Understanding,* Volume 83, pp. 236-274.

Kho, D., 2014. *Teknik Elektronika.* [Online]
Available at: https://teknikelektronika.com/fungsi-pengertian-speaker-prinsip-kerja-speaker/ [Accessed 18 April 2018].

Liao, S. et al., 2007. *Learning Multi-scale Block Local Binary Patterns for Face Recognition,* Beijing: Institute of Automation, Chinese Academy of Sciences.

Lienhart, R. & Maydt, J., 2002. *An Extended Set of Haar-like Features for Rapid Object Detection,* Santa Clara: Intel Labs.

Lopez, L. S., 2010. *Local Binary Patterns applied to Face Detection and Recognition ,* Barcelona: Telecom BCN.

Madenda, S., 2015. *Pengolahan Citra Dan Video Digital.* 1st ed. Yogyakarta: Andi Yogyakarta.

Mahesh, G., 2016. *Easy Learning Spot.* [Online]
Available at: http://easylearningspot.com/dip-prog-convert-rgb-gray-binary/ [Accessed 7 April 2018].

Man, J., 2016. *Element 14.* [Online]
Available at: https://www.element14.com/community/docs/DOC-80899/l/raspberry-pi-3-model-b-technical-specifications [Accessed 5 April 2018].

Mulyono, T., Adi, K. & Gernowo, R., 2012. Sistem Pengenalan Wajah Dengan Metode Eigenface Dan Jaringan Syaraf Tiruan (JST). *Berkala Fisika,* XV(1), pp. 15-20.

Munandar, A., 2013. *Les Elektronika.* [Online]
Available at: http://www.leselektronika.com/2013/03/membuat-driver-motor-dengan-ic-l293d.html [Accessed 25 April 2018].

Nugroho, S. & Harjoko, A., 2004. *Sistem Pendeteksi Wajah Manusia Pada Citra Digital,* Yogyakarta: Program Studi Ilmu Komputer Pascasarjana UGM.

OpenCV, O., 2018. *OpenCV.* [Online] Available at: https://opencv.org/ [Accessed 23 April 2018].

Pamungkas, A., 2014. *Pemrograman Matlab.* [Online] Available at: https://pemrogramanmatlab.com/2017/07/25/thresholding_citra/ [Accessed 6 April 2018].

Pavani, S.-K., Delgado, D. & Frangi, A. F., 2010. Haar-Like Features With Optimally Weighted Rectangles For Rapid Object Detection. *Pattern Recognition,* Volume 43, pp. 160-172.

Permana, A. E. & Dwiyono, 2015. Sistem Buka Tutup Pintu Otomatis Menggunakan Jam Tangan Berbasis Mikrokontroller. *Go Infotech,* XXI(1), pp. 13-17.

Pietikainen, M., Hadid, A., Zhao, G. & Ahonen, T., 2011. *Computer Vision Using Local Binary Patern.* 1st ed. British: British Library Cataloguing in Publication Data.

Purwanto, P., Dirgantoro, B. & Jati, A. N., 2015. Implementasi Face Identification Dan Face Recognition Pada Kamera Pengawas Sebagai Pendeteksi Bahaya. *e-Proceeding of Engineering,* II(1), pp. 718-724.

Putra, A. E., 2012. *UGM Staff Website.* [Online] Available at: http://agfi.staff.ugm.ac.id/blog/index.php/2012/08/mengenal-raspberry-pi/ [Accessed 5 April 2018].

Putra, D., 2010. *Pengolahan Citra Digital.* 1st ed. Yogyakarta: Andi Yogyakarta.

Python, O., n.d. *Python Software Foundation.* [Online] Available at: https://www.python.org/community/logos/

Rahajoeningroem, T. & Wahyudin, 2013. Home Security System with Monitoring using Cellular Phone Network. *Telekomunikasi, Kendali dan Elektronika Terapan (TELEKONTRAN),* I(1), pp. 24-32.

Raspberry Pi, O., 2014. *The Raspberry Pi Foundation.* [Online] Available at: https://www.raspberrypi.org/documentation/usage/gpio/README.md [Accessed 5 April 2018].

Raspberry Pi, O., 2018. *Raspberry Pi Visual Identity Guidelines.* 3.2 ed. United Kingdom: The Raspberry Pi Foundation.

Sajati, H., 2015. *STT Adisutjipto.* [Online] Available at: http://jati.stta.ac.id/2015/09/deteksi-obyek-menggunakan-haar-cascade.html [Accessed 7 April 2018].

Saputri, W. E., 2017. *Kelas Desain.* [Online] Available at: http://kelasdesain.com/apa-itu-rgb/ [Accessed 6 April 2018].

Sehman, 2015. *Penerapan Face Recognition dengan Metode Eigenface pada Intelligent Car Security.* Surabaya, IdeaTech, pp. 342-348.

Sepritahara, 2012. *Sistem Pengenalan Wajah (Face Recognition) Menggunakan Metode Hidden Markov Model (HMM),* Depok: Universitas Indonesia Library.

Setiawan, A., 2011. *Mikrokontroller ATMEGA 8535 & ATMEGA16 menggunakan BASCOM-AVR.* 1st ed. Yogyakarta: Andi Yogyakarta.

Singh, V., Shokeen, V. & Singh, B., 2013. Face Detection By Haar Cascade Classifier With Simple And Complex Backgrounds Images Using OpenCV Implementation. *International Journal of Advanced Technology in Engineering and Science (IJATES),* I(12), pp. 33-38.

Sung, K.-K., 1996. *Learning And Example Selection For Object And Pattern Detection,* Massachusetts: Massachusetts Institute of Technology Artificial Intelligence Laboratory.

Surjono, H. D., 1996. Eksperimen Pengiriman Sinyal Televisi Dengan Pemancar TV Dan CCTV Serta Pemanfaatannya Dalam Pendidikan. *Penelitian Tindakan Kelas (PTK),* V(7), pp. 37-43.

Sutanto, H., 1998. [Online]
Available at: http://mikrokontroler.tripod.com/6805/bab1.htm
[Accessed 22 April 2018].

Syahrul, 2014. *Pemrograman Mikrokontroller AVR Bahasa Assembly dan C.* 1st ed. Yogyakarta: INFORMATIKA.

Viola, P. & Jones, M., 2001. *Rapid Object Detection Using A Boosted Cascade Of Simple Features.* Cambridge, Accepted Conference On Computer Vision And Pattern Recognition 2001.

Wurdianarto, S. R., Novianto, S. & Rosyidah, U., 2014. Perbandingan Euclidean Distance Dengan Canberra Distance Pada Face Recognition. *Techno COM,* 13(1), pp. 31-37.

Xie, L., 2013. *MathWorks.* [Online] Available at: https://au.mathworks.com/matlabcentral/fileexchange/44648-hysteresis-thresholding-for-3d-images--or-2d- [Accessed 7 April 2018].

Zuhal, 1990. *Dasar Teknik Tenaga Listrik Dan Elektronika Daya.* 1st ed. Jakarta: Gramedia Pustaka Utama.