RANCANG BANGUN PENGONTROLAN ROBOT KAMERA BERBASIS ANDROID

SKRIPSI

Disusun Oleh :

IWAN PERNANDO PANJAITAN

NIM. 0812516

JURUSAN TEKNIK ELEKTRO S-1

KONSENTRASI TEKNIK KOMPUTER

FAKULTAS TEKNOLOGI INDUSTRI

INSTITUT TEKNOLOGI NASIONAL MALANG

2014

RANCANG BANGUN PENGONTROLAN ROBOT KAMERA BERBASIS ANDROID

SKRIPSI

Disusun Oleh :

IWAN PERNANDO PANJAITAN
NIM. 0812516

JURUSAN TEKNIK ELEKTRO S-1

KONSENTRASI TEKNIK KOMPUTER DAN INFORMATIKA

FAKULTAS TEKNOLOGI INDUSTRI

INSTITUT TEKNOLOGI NASIONAL MALANG

2014

LEMBAR PERSETUJUAN

RANCANG BANGUN PENGONTROLAN ROBOT KAMERA BERBASIS ANDROID

SKRIPSI

Disusun dan Diajukan Sebagai melengkapi dan memenuhi persyaratan guna mencapai Gelar Sarjana Teknik Komputer Strata Satu (S-1)

Disusun oleh :

IWAN PERNANDO PANJAITAN
NIM. 0812516

Mengetahui,
Ketua Jurusan Teknik Elektro S-1

M. Ibrahim Ashari, ST, MT
NIP.P. 1030100358

Diperiksa dan Disetujui

Dosen Pembimbing I	Dosen Pembimbing II
Ir. Yusuf Ismail Nakhoda, MT	**Dr. Eng. Aryuanto Soetedjo, ST,MT**
NIP.Y.1018800189	**NIP.Y.1030800417**

JURUSAN TEKNIK ELEKTRO S-1
KONSENTRASI TEKNIK KOMPUTER
FAKULTAS TEKNOLOGI INDUSTRI
INSTITUT TEKNOLOGI NASIONAL MALANG
2014

SURAT PERNYATAAN ORISINALITAS

Yang bertanda tangan di bawah ini :

Nama : IWAN PERNANDO PANJAITAN

NIM : 08.12.516

Program Studi : TEKNIK ELEKTRO S-1

Konsentrasi : TEKNIK KOMPUTER

Dengan ini menyatakan bahwa Skripsi yang saya buat adalah hasil karya sendiri, tidak merupakan plagiasi dari karya orang lain. Dalam Skripsi ini tidak memuat karya orang lain, kecuali dicantumkan sumbernya sesuai dengan ketentuan yang berlaku.

Demikian surat pernyataan ini saya buat, dan apabila di kemudian hari ada pelanggaran atas surat pernyataan ini, saya bersedia menerima sangsinya.

Malang, Februari 2014

Yang membuat Pernyataan,

Iwan Pernando

0812516

RANCANG BANGUN PENGONTROLAN ROBOT KAMERA BERBASIS ANDROID

Iwan Pernando Panjaitan

08.12.516

Email: iwslash@gmail.com

Jurusan Teknik Elektro S-1, Konsentrasi Teknik Komputer

Fakultas Teknologi Industri, Institut Teknologi Nasional Malang

Jl. Raya Karanglo Km 2 Malang

Abstrak

Penggunaan sistem operasi Android pada Smartphone semakin bertambah tiap tahunnya, banyak hal yang dapat dilakukan dengan smartphone, salah satunya pada pengontrolan robot kamera. Smartphone android memiliki Wifi dan kamera yang dapat dikembangkan pada pengontrolan robot jarak jauh. Pemanfaatan Wifi pada smartphone android banyak dikembangkan untuk komunikasi antar device yang berbeda, baik satu arah maupun dua arah secara real time. Kamera yang dimiliki smartphone juga dapat digunakan memonitoring pergerakan robot. Kedua fitur ini dapat dimanfaatkan dalam rancang bangun robot kamera dengan cara, komputer mengirimkan sinyal Wifi kepada smarphone, lalu smartphone akan memproses data tersebut untuk menggerakan motor dan secara real time smartphone mengirimkan data kamera pada komputer sebagai monitoring robot. Aplikasi ini diharapkan bisa memberikan masukan bagi perkembangan pengontrolan robot.

Kata kunci *: Smartphone Android, Wifi, Kamera, Pengontrol robot.*

Abstract

The use of the Android operating system on smartphones is increasing each year, many things to do with a smartphone, one of them on "camera controlling a robot that has a camera". Android smartphone has Wi-Fi and a camera that can be developed in remote robot control. Exploiting the Wifi on android smartphones are developed for communication between different devices, either one-way or two-way real time. Owned a smartphone camera can also be used monitor the movement of the robot. Both of these features can be exploited in the design of robots camera, the computer methods transmit WiFi signals to the smarphone, then the smartphone will process the data to drive the motor in real time and transmit data smartphone cameras monitoring the computer as a robot. This application is expected to provide input for the development of robot control.

Keywords: Android Smartphone, Accelometer sensor, Bluetooth, Wheeled Robot Controller.

KATA PENGANTAR

Dengan mengucap syukur kehadirat Tuhan Yang Maha Esa yang dengan segala Kasih dan Anugerah–Nya, telah memberikan kekuatan, kesabaran, bimbingan dan perlindungan sehingga penulis dapat menyelesaikan laporan skripsi dengan judul:

" RANCANG BANGUN PENGONTROLAN ROBOT KAMERA BERBASIS ANDROID"

Pembuatan skripsi ini disusun guna memenuhi syarat akhir kelulusan pendidikan jenjang Strata I di Institut Teknologi Nasional Malang. Dalam penyusunan skripsi ini penulis banyak mendapat bantuan baik moril maupun materiil, saran dan dorongan semangat dari berbagai pihak, untuk itu penulis mengucapkan terima kasih kepada :

1. Bapak Ir. Soeparno Djiwo, MT. selaku rektor ITN Malang
2. Bapak Ir. Sidik Noertjahjono, MT. selaku Dekan Fakultas Teknologi Industri.
3. Bapak Ir. Yusuf Ismail Nakhoda, MT. selaku Ketua Jurusan Teknik Elektro S-1 ITN Malang.
4. Bapak Dr. Eng. Aryuanto Soetedjo, ST, MT. selaku Sekretaris Jurusan Teknik Elektro S-1 ITN Malang.
5. Bapak M. Ibrahim Ashari, ST, MT. selaku Dosen Pembimbing I.
6. Bapak Ir. Yusuf Ismail Nakhoda, MT. selaku Dosen Pembimbing II.
7. Dan semua pihak yang telah membantu penulis dalam menyelesaikan skripsi ini yang tidak bisa penulis sebutkan satu persatu.

Penulis menyadari bahwa laporan ini masih banyak yang perlu disempurnakan. Oleh sebab itu kritik dan saran yang membangun sangat diharapkan. Akhir kata, penulis mohon maaf kepada semua pihak bilamana selama penyusunan skripsi ini penyusun membuat kesalahan secara tidak sengaja dan semoga skripsi ini dapat bermanfaat bagi kita semua.

Malang, Februari 2014

penulis

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

BAB I
PENDAHULUAN

1.1 LATAR BELAKANG

Perkembangan teknologi saat ini yang semakin pesat membuat manusia dapat melakukan sesuatunya dengan mudah dan cepat, salah satu yang menjadi trend saat ini adalah smartphone. Smartphone bukan berfiturkan untuk telpon dan sms saja tetapi sudah memiliki fitur GPS, Accelerometer, Touchscreen, Kamera , dll. Beragam vendor yang bersaing dibidang smartphone ini membuat perkembangan OS (Operating System) masing-masing vendor berbeda, salah satu yang terlaris saat ini adalah smartphone bersistem operasi Android.

Dengan pengembangan smartphone yang dibangun pada sistem operasi proprietary yang membatasi pengembangan dan penyebaran aplikasi pihak ketiga, Android menawarkan dan membuka alternatif. Pengembang Android bebas untuk membuat aplikasi yang mengambil keuntungan penuh dari perangkat mobile dan mendistribusikan mereka ke pasaran dan berdampak minat pengembang perangkat Android telah meledak sebagai penjualan ponsel terus tumbuh. Pada tahun 2009 dan bagian-bagian awal tahun 2010 lebih dari 20 handset Android telah dirilis dari OEM (Original Equipment Manufacturer).

Robot merupakan salah satu teknologi yang dapat membantu manusia. Berbagai bidang telah memakai robot untuk mempercepat kinerja mereka, mulai dari bidang industry, informatika, geodesi dll. Komunikasi data pada system robot juga bervariatif, mulai dari kabel serial, bluetooth,wifi hingga melalui web.Robot adalah alat yang digunakan oleh manusia sebagai pengganti dalam melakukan pekerjaannya. Robot mengalami perkembangan yang sangat pesat dan sangat disenangi oleh mauusia, karena robot bisa dijadikan teman dalam melakukan pekerjaan sehari – hari. Pada tugas akhir ini penulis ingin mengabungkan 2 teknologi ini agar dapat berkomunikasi diatas sistem operasi android dan dapat memberikan instruksi ataupun pengolah data robot itu.

1.2 Rumusan Masalah

Dari later belakang masalah yang telah penulis paparkan, maka penulis mencoba untuk mengembangkan sebuah sistem yang mudah dan efisien. Adapun beberapa permasalahan yang ada yaitu :

1. Bagaimana melakukan koneksi Wifi antara robot dengan smartphone berbasis android.
2. Bagaimana membuat program di Smartphone supaya bisa menerima perintah dari aplikasi PC yang ada di Smartphone Android kemudian mengendalikan pergerakan robot.

1.3 Tujuan

Adapun tujuan dari pembuatan tugas akhir ini :

1. Membantu kinerja mikrokontroller yang memiliki keterbatasan memori.
2. Mengontrol robot dengan smartphone beroperasi sistem Android.
3. Mengembangkan system operasi android dalam bidang robotika.

1.4 Manfaat

Adapun manfaat penulisan skripsi ini adalah :

1. Dapat menjadi pengganti alat control lama yang masih menggunakan joystick yang terhubung dengan kabel.
2. Untuk perkembangan robotika di Institut Teknologi Nasional Malang.

1.5 Batasan Makalah

Untuk mencapai tujuan penyelesaian skripsi ini secara maksimal, maka diperlukan batasan masalah yang diharapkan agar permasalahan tidak meluas dan tetap fokus pada tujuan utama. Adapun batasan-batasan masalah pada skripsi ini yaitu :

1. Robot yang digunakan adalah robot beroda/*mobile robot*.
2. Aplikasi hanya dapat dijalankan pada *smartphone* berbasis *Android* yang memiliki fungsi *Wifi* dan *kamera*.
3. Tidak membahas konsep jaringan secara detail.
4. Tidak membahas *hardware* secara detail.

1.6 Metodologi Penelitian

Adapun metode penelitian yang digunakan dalam pembuatan perangkat lunak ini adalah sebagai berikut:

1. **Studi literatur**

   Pengumpulan data yang dilakukan dengan mencari bahan-bahan kepustakaan, referensi, dan dari berbagai sumber sebagai landasan teori yang ada hubungannya dengan permasalahan yang dijadikan objek penelitian.

2. **Analisa Kebutuhan Aplikasi**

   Data dan informasi yang telah diperoleh akan dianalisa agar dihasilkan kerangka global yang bertujuan untuk mendefinisikan kebutuhan sistem dimana natinya akan digunakan sebagai acuan perancangan sistem.

3. **Perancangan dan Implementasi**

   Berdasarkan data dan informasi yang telah diperoleh serta analisa kebutuhan untuk membangun sistem ini, akan dibuat rancangan kerangka global yang menggambarkan mekanisme dari sistem yang akan dibuat dan diimplementasikan ke dalam sistem.

4. **Eksperimen dan Evaluasi**

   Pada tahap ini, sistem yang telah selesai dibuat akan diuji coba, yaitu pengujian berdasarkan fungsionalitas program dan penyempurnaan sistem jika diperlukan.

5. **Penulisan laporan tugas akhir.**

   Mengintegrasikan sistem antara hardware dengan software, kemudian dilakukan pengujian dan analisis terhadap hasil yang telah didapatkan

1.7 Sistematika Penulisan

Untuk mempermudah dan memahami pembahasan penulisan skripsi ini, maka sistimatika penulisan disusun sebagai berikut :

BAB I : **PENDAHULUAN**

Bab ini merupakan bagian pendahuluan dimana akan tercakup secara umum mengenai latar belakang penulisan laporan, ruang lingkup karya tulis skripsi ini, tujuan dan manfaat, metodologi yang

dipakai dalam penyusunan laporan dan sistematika penulisan yang digunakan.

BAB II : LANDASAN TEORI

Bab ini berisi tentang teori - teori yang mendukung dan berhubungan dengan judul penulisan skripsi

BAB III : ANALISA DAN PERANCANGAN SISTEM

Bab ini berisi mengenai analisa kebutuhan sistem baik software maupun hardware yang di perlukan untuk membuat kerangka global yang menggambarkan mekanisme dari sistem yang akan di buat.

BAB IV : IMPLEMENTASI DAN PENGUJIAN SISTEM

Bab ini berisi tentang implementasi dari perancangan sistem yang telah dibuat serta pengujian terhadap sistem tersebut.

BAB V : PENUTUP

Merupakan bab terakhir yang memuat intisari dari hasil pembahasan yang berisikan kesimpulan dan saran yang dapat digunakan sebagai pertimbangan untuk pengembangan penulisan selanjutnya.


INSTITUT TEKNOLOGI NASIONAL
ITN
MALANG

BAB II
TINJAUAN PUSTAKA

2.1 Sistem Operasi Android

Gambar 2.1 Sistem Operasi Android

Android adalah sistem operasi berbasis Linux yang dirancang untuk perangkat seluler layar sentuh seperti telepon pintar dan komputer tablet. Android awalnya dikembangkan oleh Android, Inc., dengan dukungan finansial dari Google, yang kemudian membelinya pada tahun 2005. Sistem operasi ini dirilis secara resmi pada tahun 2007, bersamaan dengan didirikannya Open Handset Alliance, konsorsium dari perusahaan-perusahaan perangkat keras, perangkat lunak, dan telekomunikasi yang bertujuan untuk memajukan standar terbuka perangkat seluler. Ponsel Android pertama mulai dijual pada bulan Oktober 2008.

Antarmuka pengguna Android didasarkan pada manipulasi langsung, menggunakan masukan sentuh yang serupa dengan tindakan di dunia nyata, seperti menggesek, mengetuk, mencubit, dan membalikkan cubitan untuk memanipulasi obyek di layar. Android adalah sistem operasi dengan sumber terbuka, dan Google merilis kodenya di bawah Lisensi Apache. Kode dengan sumber terbuka dan lisensi perizinan pada Android memungkinkan perangkat lunak untuk dimodifikasi secara bebas dan didistribusikan oleh para pembuat perangkat, operator nirkabel, dan pengembang aplikasi. Selain itu, Android memiliki sejumlah besar komunitas pengembang aplikasi (apps) yang memperluas fungsionalitas perangkat, umumnya ditulis dalam versi kustomisasi bahasa pemrograman Java. Pada bulan Oktober 2012, ada sekitar 700.000 aplikasi yang tersedia untuk Android, dan sekitar 25 juta aplikasi telah diunduh dari Google Play, toko aplikasi utama Android.

Sebuah survey pada bulan April-Mei 2013 menemukan bahwa Android adalah platform paling populer bagi para pengembang, digunakan oleh 71% pengembang aplikasi seluler.

Faktor-faktor di atas telah memberikan kontribusi terhadap perkembangan Android, menjadikannya sebagai sistem operasi telepon pintar yang paling banyak digunakan di dunia, mengalahkan Symbian pada tahun 2010. Android juga menjadi pilihan bagi perusahaan teknologi yang menginginkan sistem operasi berbiaya rendah, bisa dikustomisasi, dan ringan untuk perangkat berteknologi tinggi tanpa harus mengembangkannya dari awal. Akibatnya, meskipun pada awalnya sistem operasi ini dirancang khusus untuk telepon pintar dan tablet, Android juga dikembangkan menjadi aplikasi tambahan di televisi, konsol permainan, kamera digital, dan perangkat elektronik lainnya. Sifat Android yang terbuka telah mendorong munculnya sejumlah besar komunitas pengembang aplikasi untuk menggunakan kode sumber terbuka sebagai dasar proyek pembuatan aplikasi, dengan menambahkan fitur-fitur baru bagi pengguna tingkat lanjut atau mengoperasikan Android pada perangkat yang secara resmi dirilis dengan menggunakan sistem operasi lain.

Pada November 2013, Android menguasai pangsa pasar telepon pintar global, yang dipimpin oleh produk-produk Samsung, dengan persentase 64% pada bulan Maret 2013. Pada Juli 2013, terdapat 11.868 perangkat Android berbeda dengan beragam versi. Keberhasilan sistem operasi ini juga menjadikannya sebagai target ligitasi paten "perang telepon pintar" antar perusahaan-perusahaan teknologi. Hingga bulan Mei 2013, total 900 juta perangkat Android telah diaktifkan di seluruh dunia, dan 48 miliar aplikasi telah dipasang dari Google Play. Pada tanggal 3 September 2013, 1 miliar perangkat Android telah diaktifkan.

2.1.1 Fitur Antarmuka Android

Antarmuka pengguna pada Android didasarkan pada manipulasi langsung, menggunakan masukan sentuh yang serupa dengan tindakan di dunia nyata, misalnya menggesek (*swiping*), mengetuk (*tapping*), dan mencubit (*pinching*), untuk memanipulasi obyek di layar. Masukan pengguna direspon dengan cepat dan juga tersedia antarmuka sentuh layaknya permukaan air, seringkali menggunakan kemampuan getaran perangkat untuk memberikan umpan balik haptik kepada pengguna. Perangkat keras internal seperti akselerometer, giroskop, dan sensor proksimitas digunakan oleh beberapa aplikasi untuk

merespon tindakan pengguna, misalnya untuk menyesuaikan posisi layar dari potret ke lanskap, tergantung pada bagaimana perangkat diposisikan, atau memungkinkan pengguna untuk mengarahkan kendaraan saat bermain balapan dengan memutar perangkat sebagai simulasi kendali setir.

Ketika dihidupkan, perangkat Android akan memuat pada layar depan (homescreen), yakni navigasi utama dan pusat informasi pada perangkat, serupa dengan desktop pada komputer pribadi. Layar depan Android biasanya terdiri dari ikon aplikasi dan widget; ikon aplikasi berfungsi untuk menjalankan aplikasi terkait, sedangkan widget menampilkan konten secara langsung dan terbarui otomatis, misalnya prakiraan cuaca, kotak masuk surel pengguna, atau menampilkan tiker berita secara langsung dari layar depan. Layar depan bisa terdiri dari beberapa halaman, pengguna dapat menggeser bolak balik antara satu halaman ke halaman lainnya, yang memungkinkan pengguna Android untuk mengatur tampilan perangkat sesuai dengan selera mereka. Beberapa aplikasi pihak ketiga yang tersedia di Google Play dan di toko aplikasi lainnya secara ekstensif mampu mengatur kembali tema layar depan Android, dan bahkan bisa meniru tampilan sistem operasi lain, misalnya Windows Phone. Kebanyakan produsen telepon seluler dan operator nirkabel menyesuaikan tampilan perangkat Android buatan mereka untuk membedakannya dari pesaing mereka.

Di bagian atas layar terdapat status bar, yang menampilkan informasi tentang perangkat dan konektivitasnya. Status bar ini bisa "ditarik" ke bawah untuk membuka layar notifikasi yang menampilkan informasi penting atau pembaruan aplikasi, misalnya surel diterima atau SMS masuk, dengan cara tidak mengganggu kegiatan pengguna pada perangkat. Pada versi awal Android, layar notifikasi ini bisa digunakan untuk membuka aplikasi yang relevan, namun setelah diperbarui, fungsi ini semakin disempurnakan, misalnya kemampuan untuk memanggil kembali nomor telepon dari notifikasi panggilan tak terjawab tanpa harus membuka aplikasi utama. Notifikasi ini akan tetap ada sampai pengguna melihatnya, atau dihapus dan di nonaktifkan oleh pengguna.

2.1.2 Fitur Aplikasi Android

Android memungkinkan penggunanya untuk memasang aplikasi pihak ketiga, baik yang diperoleh dari toko aplikasi seperti Google Play, Amazon Appstore, ataupun dengan mengunduh dan memasang berkas APK dari situs pihak ketiga. Di Google Play, pengguna

bisa menjelajah, mengunduh, dan memperbarui aplikasi yang diterbitkan oleh Google dan pengembang pihak ketiga, sesuai dengan persyaratan kompatibilitas Google. Google Play akan menyaring daftar aplikasi yang tersedia berdasarkan kompatibilitasnya dengan perangkat pengguna, dan pengembang dapat membatasi aplikasi ciptaan mereka bagi operator atau negara tertentu untuk alasan bisnis. Pembelian aplikasi yang tidak sesuai dengan keinginan pengguna dapat dikembalikan dalam waktu 15 menit setelah pengunduhan. Beberapa operator seluler juga menawarkan tagihan langsung untuk pembelian aplikasi di Google Play dengan cara menambahkan harga pembelian aplikasi pada tagihan bulanan pengguna. Pada bulan September 2012, ada lebih dari 675.000 aplikasi yang tersedia untuk Android, dan perkiraan jumlah aplikasi yang diunduh dari Play Store adalah 25 miliar.

Aplikasi Android dikembangkan dalam bahasa pemrograman Java dengan menggunakan kit pengembangan perangkat lunak Android (SDK). SDK ini terdiri dari seperangkat perkakas pengembangan,[61] termasuk debugger, perpustakaan perangkat lunak, emulator handset yang berbasis QEMU, dokumentasi, kode sampel, dan tutorial. Didukung secara resmi oleh lingkungan pengembangan terpadu (IDE) Eclipse, yang menggunakan plugin Android Development Tools (ADT). Perkakas pengembangan lain yang tersedia di antaranya adalah Native Development Kit untuk aplikasi atau ekstensi dalam C atau C++, Google App Inventor, lingkungan visual untuk pemrogram pemula, dan berbagai kerangka kerja aplikasi web seluler lintas platform.

2.1.3 Fitur Pengelolaan Memori Android

Android dikembangkan secara pribadi oleh Google sampai perubahan terbaru dan pembaruan siap untuk dirilis, dan informasi mengenai kode sumber juga mulai diungkapkan kepada publik. Kode sumber ini hanya akan berjalan tanpa modifikasi pada perangkat tertentu, biasanya pada seri Nexus. Ada binari tersendiri yang disediakan oleh produsen agar Android bisa beroperasi.

Logo Android yang berwarna hijau awalnya dirancang untuk Google pada tahun 2007 oleh desainer grafis Irina Blok. Tim desain ditugaskan dengan sebuah proyek untuk membuat sebuah ikon universal yang mudah dikenali dengan menyertakan ikon robot secara spesifik dalam desain akhir. Setelah sejumlah perkembangan desain yang didasarkan pada tema-tema fiksi ilmiah dan film luar angkasa, tim akhirnya mendapat inspirasi dari

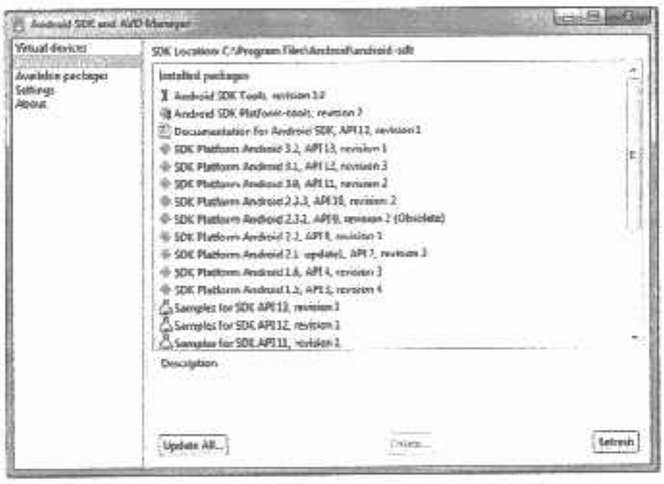
simbol manusia yang terdapat di pintu toilet, dan memodifikasi bentuknya menjadi bentuk robot. Karena Android adalah perangkat lunak sumber terbuka, disepakati bahwa logo tersebut juga harus terbuka, dan sejak diluncurkan, logo hijau tersebut telah didesain ulang kembali dalam berbagai variasi yang tak terhitung jumlahnya[1].

2.2 Android Scripting Environment (ASE)

Android Scripting Environtment(ASE) memudahkan bahasa scripting untuk Android dengan memungkinkan kita untuk mengedit dan mengeksekusi script langsung pada perangkat Android. Script ini memiliki akses ke banyak API tersedia untuk penuh aplikasi Android, tetapi dengan antarmuka sangat sederhana yang membuatnya mudah untuk menyelesaikan sesuatu.

Script dapat dijalankan secara interaktif dalam terminal, di background, atau melalui Lokal. Berikut ini adalah software / tool yang akan digunakan pada pembuatan robot ini :

1) Android SDK



Gambar 2.2 Android Sdk

Android SDK diperlukan membuat aplikasi android yang kemudian akan di compress menjadi format .APK pada ponsel. Pada Android SDK terdapat emulator, kompiler, console, plugin dan sampel yang akan sangat berguna bagi developer android. Pada table 2.1 menjelaskan perkembangan sejarah SDK Android.

[1] Wikipedia, "Pengertian Sistem Operasi Android" dalam http://id.wikipedia.org/wiki/Android_%28sistem_operasi%29

Table 2.1 Android SDK

Versi	Nama kode	Tanggal rilis	Level API	Distribusi
4.4	KitKat	31 Oktober 2013	19	
4.3.x	Jelly Bean	24 Juli 2013	18	2,3%
4.2.x	Jelly Bean	13 November 2012	17	12,5%
4.1.x	Jelly Bean	9 Juli 2012	16	37,3%
4.0.3–4.0.4	Ice Cream Sandwich	16 Desember 2011	15	19,8%
3.2	Honeycomb	15 Juli 2011	13	0,1%
3.1	Honeycomb	10 Mei 2011	12	0,0%
2.3.3–2.3.7	Gingerbread	9 Februari 2011	10	26,3%
2.3–2.3.2	Gingerbread	6 Desember 2010	9	0%
2.2	Froyo	20 Mei 2010	8	1,7%
2.0–2.1	Eclair	26 Oktober 2009	7	0%
1.6	Donut	15 September 2009	4	0%
1.5	Cupcake	30 April 2009	3	0%

2) Library IOIO (IOIOlib)

IOIOLib adalah kumpulan perpustakaan, untuk Android dan untuk PC, yang memungkinkan aplikasi Anda untuk mengontrol papan IOIO. IOIOlib mengekspos seperangkat interface Java, mencakup berbagai fitur dari modul. Ketika membangun aplikasi, IOIOLib akan membuat dormat ke dalam bentuk jar atau Apk.

Dua library tambahan, *IOIOLibBT* dan *IOIOLibAccessory* saling melengkapi *IOIOLibAndroid* dan menambahkan Bluetooth dan fungsi Android *Open Accessory* koneksi mungkin untuk IOIO, masing-masing. Alasan mengapa mereka terpisah adalah bahwa perpustakaan *IOIOLibAndroid* akan bekerja pada setiap versi Android (sedini V1.5), sementara Bluetooth telah diperkenalkan di Android v2.x dan *Open Accessory* hanya tersedia di pilih Android 2.3.4.

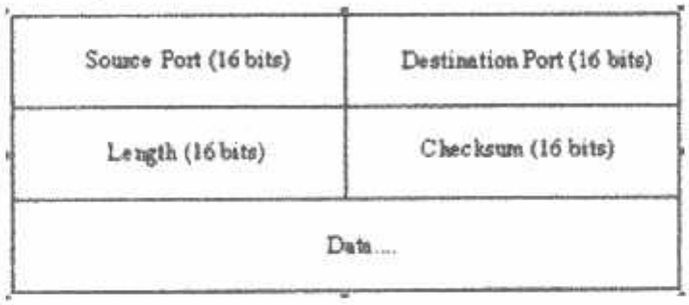
3) Java Slick2D Library

Slick-Util adalah sebuah perpustakaan kecil untuk memungkinkan Anda untuk memuat berbagai format gambar, suara dan font. Dengan sebuah forum aktif dan fitur baru dalam pengembangan Slick2D memudahkan pengembangan bagi developer.

2.3 Konsep UDP

TCP merupakan protokol *connection-oriented*. Ada kalanya dimana protokol berorientasi *connectionless* dibutuhkan, makanya UDP digunaka. UDP digunakan untuk

trivial file transfer protocol (TFTP) dan *remote call procedure* (RCP). Komunikasi *connectionless* tidak mendukung reliabilitas, artinya tidak ada informasi yang yang diterima oleh mesin pengirim yang mengindikasikan data diterima oleh mesin penerima dengan benar. Protokol *connctionless* juga tidak memiliki kemampuan untuk melakukan recover terhadap data yang mengalami error. UDP lebih sederhana dinbanding TCP. UDP berhubungan langsung dengan IP tanpa adanya mekanisme *flow control* dan *error-recovery*. *Header message* UDP lebih sederhana dibandingkan TCP. Sebagaimana terlihat pada gambar . *Field padding* dapat ditambahkan ke datagram untuk memastikan bahwa message terdiri atas multiple 16-bit.



Gambar 2.3 Field Udp

Field-fieldnya adalah sebagai berikut:

- **Source port:** field optional dengan nomor port. Jika tidak ada nomor port yang ditentukan, field tersebut diset menjadi 0.
- **Destination port:** nomor port mesin tujuan.
- **Length:** panjang datagram, termasuk header dan data.
- **Checksum:** field dengan 16-bit komplement satu dari jumlah komplemen satu dari datagram, termasuk pseudohcadcwr yang sama dengan TCP.

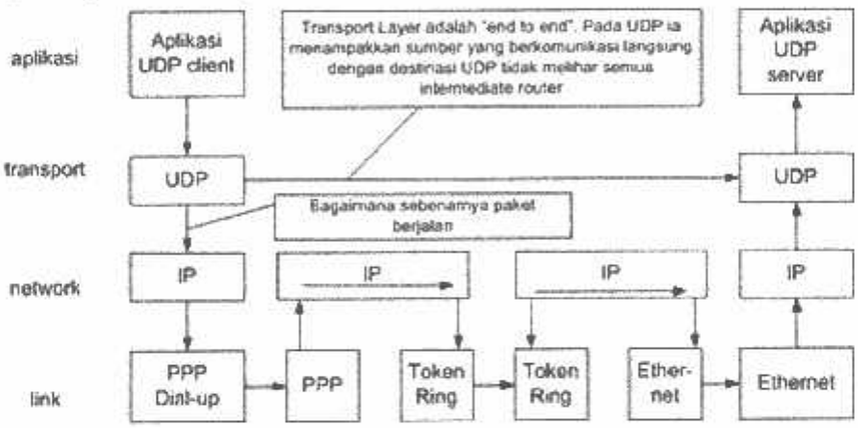
Field checksum pada UDP hanya merupakan optional, tetapi jika tidak digunakan, maka tidak akan ada checksum pada segmen data karena checksum IP hanya digunakan pada header IP. Jika checksum tidak digunakan, field ini akan diset menjadi 0.

UDP adalah protokol transport yang digunakan secara luas pada lapisan di atas IP. Seperti TCP, UDP menggunakan port dan menyediakan konektivitas *end-to-end* antara aplikasi client dan server. UDP merupakan protokol yang kecil dan efisien. Tetapi, berbeda dengan TCP, UDP tidak menjamin pengiriman – aplikasi harus mengimplementasikan mekanisme

error recovery sendiri — jika memerlukan mekanisme tersebut. Hal ini membuatnya cocok untuk beberapa aplikasi, tetapi tidak untuk beberapa yang lain.

Dalam beberapa hal, UDP mirip dengan TCP :

1. UDP adalah protokol transport : UDP hanya berhubungan dengan komunikasi antara dua end point (misalnya aplikasi client pada mesin Anda, dan aplikasi server pada mesin remote). Intermediate router tidak berhubungan dengan data UDP dalam paket yang dikirimkannya – router hanya beroperasi pada layer IP atau network lower-down.
2. UDP menggunakan port untuk membedakan antara trafic dari banyak aplikasi UDP pada mesin yang sama, dan untuk mengirim paket yang tepat ke aplikasi yang sesuai (ini disebut demultiplexing). UDP dan port-nya menyediakan interface antara program aplikasi dan layar networking IP



Gambar 2.4 Diagram Udp

UDP berbeda dari TCP dalam beberapa hal penting, karena:

1. UDP adalah "*datagram oriented*", TCP adalah "*session-oriented*". Datagram adalah paket informasi *self-contained*; UDP berhubungan dengan datagram atau paket individu yang dikirim dari client ke server, atau sebaliknya.
2. UDP adalah *connectionless*. Client tidak membangun koneksi ke server sebelum mengirim data client hanya mengirim data secara langsung.
3. UDP "tidak andal" dalam pengertian jaringan formal :
    1. Paket dapat hilang. UDP tidak dapat mendeteksinya.
    2. Program aplikasi client atau server (sebagai kebalikan TCP/IP stack sendiri) harus mendeteksi paket yang hilang dan menangani transmisi ulang, dan

lain-lain. Aplikasi sering menunggu hingga timeout habis, dan kemudian mencoba lagi.

3. Paket dapat mengalami kerusakan. Paket UDP berisi checksum semua data dalam paket. Checksum ini memungkinkan UDP mendeteksi kapan suatu paket mengalami kerusakan. Jika hal ini terjadi, maka paket tersebut dikeluarkan, dan sebagaimana biasa aplikasi yang harus mendeteksi hal ini dan melakukan transmisi ulang sepenuhnya.
4. Operasi checksum ini dapat dihentikan, dan beberapa aplikasi melakukannya untuk alasan unjuk kerja. Akan tetapi hal ini dapat berarti paket yang rusak tidak terdeteksi atau layer aplikasi harus melakukan pemeriksaan integritas data sendiri, hal ini merupakan false economy (penghematan finansial yang sebenarnya menuju pada pengeluaran yang lebih besar).
5. Karena UDP adalah *datagram-oriented* dan pada level protokol setiap paket berdiri sendiri, maka UDP tidak memiliki konsep paket sesuai urutan, yang selanjutnya berarti tidak memerlukan nomor urut pada paket tersebut.
6. Sejak pertama kali dikembangkan, TCP telah dilengkapi dengan mekanisme yang sangat canggih untuk mengendalikan kecepatan aliran dalam koneksinya, untuk menghindari kemacetan dan kehilangan paket yang berlebihan. Karena UDP hanya mengirim paket tunggal, yang berdiri sendiri, maka UDP tidak memerlukan mekanisme kontrol yang rumit. Hal ini membuat UDP lebih mudah dan lebih kecil (dalam baris data dan memori) untuk diimplementasikan, tetapi juga membuatnya tidak cocok untuk sejumlah besar data.

Jika suatu aplikasi diimplementasikan menggunakan UDP, bukannya TCP, maka aplikasi tersebut harus memiliki sendiri deteksi paket-hilang, retry, dan lain sebaginya.[2]

UDP mewarisi sifat IP, yaitu connectionless dan tidak andal. UDP sebagai layer transport sangat tipis di atas IP untuk memberikan akses aplikasi ke fasilitas networking dasar IP, tanpa menambahkan fungsionalitas tambahan yang sangat banyak selain port dan

[2] Bambang Iswanto "Definisi UDP" dalam http://blog.uad.ac.id/bambangis/2011/04/01/definisi-udp-user-datagram-protocol

checksum. (sebaliknya, TCP juga merupakan layer transport tetapi tidak melakukan banyak hal selain komunikasi paket IP dasar).

2.4 Konfigurasi Hardware

Pada pembuatan robot ini ada beberapa hardware yang dibutuhkan agar bisa berjalan dengan baik, berikut ini adalah penjelasan konfigurasi hardware yang akan digunakan :

2.4.1 Smartphone

Gambar 2.5 Smartphone Galaxy Mini

Smartphone adalah telepon genggam yang mempunyai kemampuan tingkat tinggi, kadang-kadang dengan fungsi yang menyerupai komputer. Bagi beberapa orang, telepon pintar merupakan telepon yang bekerja menggunakan seluruh piranti lunak sistem operasi yang menyediakan hubungan standar dan mendasar bagi pengembang aplikasi. Bagi yang lainnya, smartphone hanyalah merupakan sebuah telepon yang menyajikan fitur canggih seperti surel (surat elektronik), internet dan kemampuan membaca buku elektronik (e-book) atau terdapat papan ketik (baik built-in maupun eksternal) dan konektor VGA. Dengan kata lain, telepon pintar merupakan komputer mini yang mempunyai kapabilitas sebuah telepon.

Pertumbuhan permintaan akan alat canggih yang mudah dibawa kemana saja membuat kemajuan besar dalam prosesor, memori, layar dan sistem operasi yang di luar dari jalur telepon genggam sejak beberapa tahun ini.

Pada robot kamera akan digunakan smartphone dengan system operasi Android. Pada modul IOIO minimal versi yang digunakan telah mensupport mulai dari versi 1.5 (cupcake). Pada robot kamera ini kamera dan wifi akan diaktifkan untuk memproses kerja

robot. Pada table 2.2 adalah spesifikasi dari Samsung Galaxy Mini yang digunakan dalam pengerjaan robot.[3]

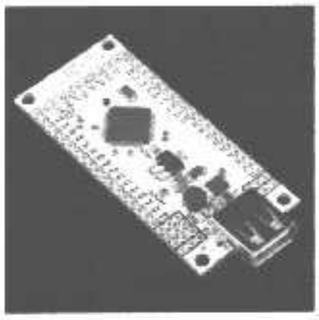
Table 2.2 Spesifikasi Samsung Galaxy Mini

General	2G Network 3G Network SIM Announced Status	GSM 850 / 900 / 1800 / 1900 HSDPA 900 / 2100 Mini-SIM 2011, January Available. Released 2011, February
Body	Dimensions Weight	110.4 x 60.8 x 12.1 mm (4.35 x 2.39 x 0.48 in) 105 g (3.70 oz)
Display	Type Size	TFT capacitive touchscreen, 256K colors 240 x 320 pixels, 3.14 inches (~127 ppi pixel density) - TouchWiz v3.0 UI
Memory	Card slot	microSD, up to 32 GB, 2 GB included
	Internal	160 MB storage, 384 MB RAM
Data	GPRS	Class 12 (4+1/3+2/2−3/1+4 slots), 32 - 48 kbps
	EDGE	Yes
	Speed	HSDPA, 7.2 Mbps
	WLAN	Wi-Fi 802.11 b/g/n, Wi-Fi hotspot
	Bluetooth	Yes, v2.1 with A2DP
	USB	Yes, microUSB v2.0
Camera	Primary	3.15 MP, 2048x1536 pixels, check quality
	Features	Geo-tagging
	Video	Yes, QVGA@15fps
	Secondary	No
Feature	OS	Android OS, v2.2 (Froyo), upgradable to v2.3 (Gingerbread)
	Chipset	Qualcomm MSM7227
	CPU	600 MHz ARMv6
	GPU	Adreno 200
	Sensors	Accelerometer, proximity, compass
	Messaging	SMS(threaded view), MMS, Email, Push Email, IM

[3] Gsm Arena "Spesifikasi Galaxy Mini" dalam http://www.gsmarena.com/samsung_galaxy_mini_s5570-3725.php

	Browser	HTML
	Radio	Stereo FM radio with RDS

2.4.2 IOIO Board



Gambar 2.6 Modul IOIO

IOIO adalah produk yang diproduksi dan dijual oleh SparkFun yang berisi mikrokontroler PIC dan antarmuka USB beberapa komponen lainnya untuk mengatur tegangan suplai dan keseluruhan proyek, baik perangkat keras dan perangkat lunak yang bersifat open source.

IOIO akan bekerja dengan berbagai ponsel Android. Kebanyakan ponsel Android dengan Android 1.5 . Jenis koneksi USB yang dibutuhkan adalah USB client. Inilah yang luas Sebagian besar ponsel Android biasa akan memiliki. Ini mengambil bentuk sedikit. Tidak seperti USB computer desktop / labtob, IOIO memakai USB mikro B konektor yang hanya support pada smartphone Android.[4]

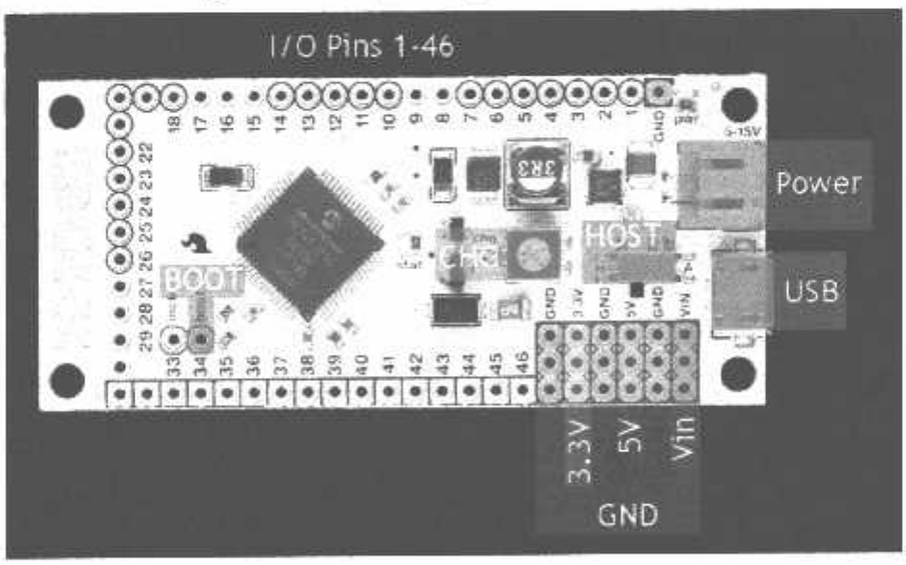
IOIO berisi komponen berikut :

1. Konektor USB (micro - AB , female) : Digunakan untuk terhubung ke host komputer , perangkat Android atau dongle Bluetooth .
2. Catu Daya listrik (2 - pin JST , female) : Digunakan untuk catu daya ke IOIO. Tegangan antara 5V - 15V harus disediakan .
3. Pin GND (10 pin) : koneksi Ground.
4. Pin VIN (3 pin) : Digunakan untuk keluaran tegangan suplai ke sirkuit , atau sebagai masukan alternatif untuk catu daya .
5. Pin 5V (3 pin) : 5V output dari regulator *on-board* , yang dapat digunakan di sirkuit.

[4] **Simon Monk**. Making Android Accessories with IOIO, Penerbit O'REILLY, Canada 2012, hal 1-20

6. Pin 3.3V (3 pin) : 3.3V dari regulator on-board , yang dapat digunakan di sirkuit .
7. I / O pin (46 pin, nomor 1-46) : Tujuan umum I / O pin. Beberapa memiliki fungsi khusus , lihat pada table 2.3 .
8. PWR LED (merah) : Led ketika IOIO mendapatkan daya .
9. STAT LED (kuning) : Tujuan umum on-board LED , di bawah kontrol aplikasi .
10. MCLR pin : Jarang digunakan. Tujuannya adalah untuk pemrograman bootloader firmware baru di papan IOIO .
11. BOOT pin : pin khusus digunakan untuk mendapatkan IOIO ke mode bootloader pada power- up .
12. Charge current trimmer (CHG): Mengatur jumlah arus yang disediakan pada baris VBUS dari USB ketika bertindak sebagai host USB . Biasanya digunakan dalam aplikasi untuk menghemat baterai .
13. Host switch: Dalam " A " mode , yang IOIO - OTG akan mendeteksi apakah harus bertindak sebagai Host atau sebagai Device, pada saat USB terpasang (micro - A atau micro - B) . Untuk mendukung kabel USB non - standar atau adapter yang menggunakan jenis micro - B , pindahkan saklar ke " H " posisi untuk merubah ke modus Host .

Pada gambar 2.6 adalah gambaran fungsi komponent modul IOIO.



Gambar 2.7 Komponen IOIO

Pada IOIO terdapat 46 pin dan beberapa pin sebagai ground dan vin yang kita bisa pakai berikut ini adalah fungsi input/output pin pada IOIO:[5]

Tabel 2.3 Fungsi Pin IOIO

Fungsi Pin	Pin Nomor
Analog in	1. 31-34, 37-46
TWI (data, clock)	1. (4, 5), (26, 25), (47, 48)
UART	3-7, 9-14, 27-32, 34-40, 45-48
5V-friendly	3-7, 10-14, 18-26, 47-48

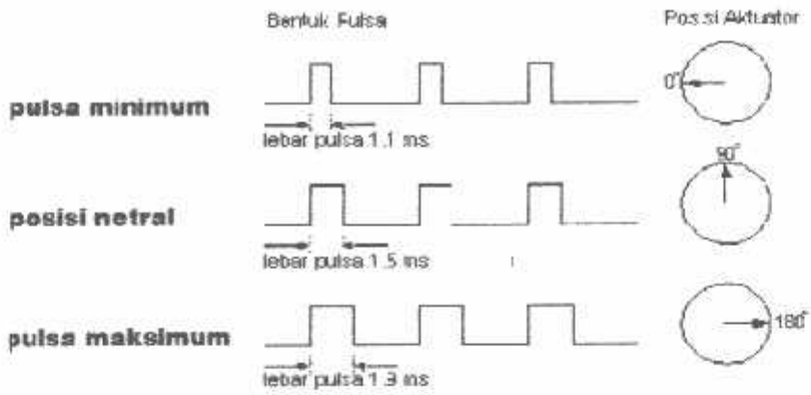
2.4.3 Motor Servo

Gambar 2.8 Motor Servo

Secara umum terdapat 2 jenis motor servo, yaitu motor servo standard dan motor servo kontinu. Servo motor tipe standar hanya mampu berputar 180 derajat. Motor servo standard sering dipakai pada sistim robotika misalnya untuk membuat " Robot Arm" (Robot Lengan). sedangkan motor servo kontinu dapat berputar sebesar 360 derajat. motor servo kontinu sering dipakai untuk *mobile robotic*. Pada badan servo tertulis tipe servo yang bersangkutan.Motor servo adalah sebuah motor dengan sistem umpan balik tertutup di mana posisi dari motor akan diinformasikan kembali ke rangkaian kontrol yang ada di dalam motor servo. Motor ini terdiri dari sebuah motor DC, serangkaian gear, potensiometer dan rangkaian kontrol. Potensiometer berfungsi untuk menentukan batas sudut dari putaran servo. Sedangkan sudut dari sumbu motor servo diatur berdasarkan lebar pulsa yang dikirim melalui kaki sinyal dari kabel motor .[6]

[5] Github, *"IOIO digital input/ output" dalam* https://github.com/ytai/ioio/wiki/DigitalIO

[6] OMAYIB "Mengenal Motor Servo" dalam http://akbarulhuda.wordpress.com/2010/04/01/mengenal-motor-servo



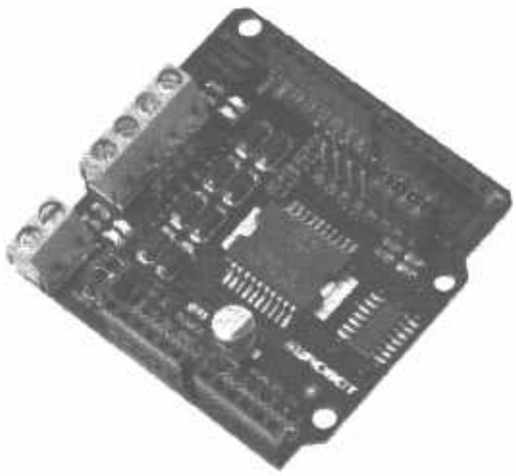
Gambar 2.9 Metode Pwm

Pengendalian gerakan batang motor servo dapat dilakukan dengan menggunakan metode PWM. (Pulse Width Modulation). Teknik ini menggunakan sistem lebar pulsa untuk mengemudikan putaran motor. Sudut dari sumbu motor servo diatur berdasarkan lebar pulsa yang dikirim melalui kaki sinyal dari kabel motor. Tampak pada gambar 2.7, pada saat diberi periode 1.5 ms maka sumbu motor akan berda pada posisi tengah, pada periode selebar 2 mS maka sudut sumbu motor akan berada pada posisi 180 derajat . Semakin lebar pulsa OFF maka akan semakin besar gerakan sumbu ke arah jarum jam dan semakin kecil pulsa OFF maka akan semakin besar gerakan sumbu ke arah yang berlawanan dengan jarum jam.

2.4.3 Motor Shield Dfrduino

Motor Shield Arduino adalah driver motor yang compatible dengan Arduino,shield ini digunakan di Arduino yang akan menggerakan motor DC. Board ini merupakan buatan DFRobot yang kompatibel dengan Arduino UNO. Dengan warna PCB hitam, dan diproduksi dengan sangat baik. Sudah tersedia 2 masukan catu daya sebagai penggeraknya. Board Arduino ini sudah memiliki 4 pin yang bisa digunakan untuk PWM dan tidak perlu lagi membeli modul *Analog to Digital Converter*, karena ada 6 channel yang bisa Anda pakai. Board kecil, namun kaya fitur. Arduino Motor Shield untuk Arduino ini menggunakan chip L298P yang memungkinkan untuk menggerakkan dua buah motor DC dengan voltase 7-12V dengan arus maksimum 2A. Shield ini dapat langsung dipasang diatas board Arduino Uno, Mega dan board Arduino compatible yg lain.[7]

[7] Arduino "How to set up your motor driver" dalam http://www.arduino.cc



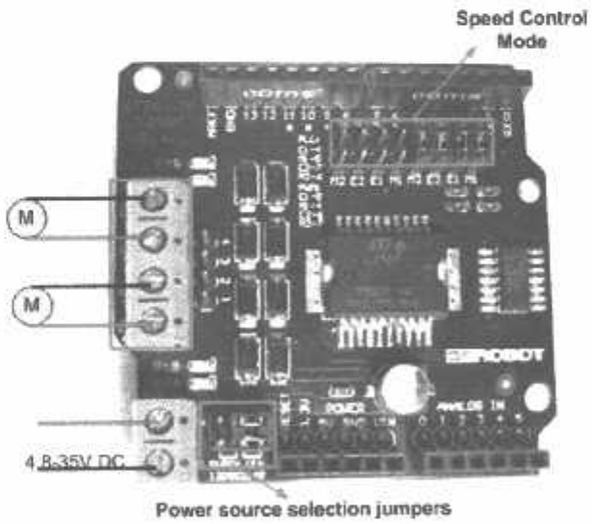
Gambar 2.10 Motor shield Arduino

Spesifikasi:

1) 2 JALUR 7-12V motor drive.
2) Arus melebihi 2A current each way.
3) Pin 5,6,7,8 are used to drive two DC motor.
4) Support PWM speed control.
5) Support PLL advance speed control.

2.4.3.1 Diagram dan Pengaturan Model Kontrol

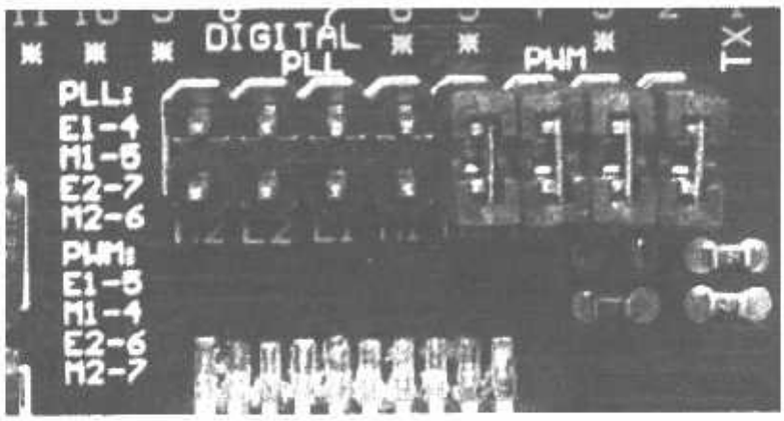
Pada gambar 2.9 adalah gambaran fungsi dari DfrFuino



Gambar 2.11 Gambaran Fungsi dari DfrFuino

2.4.3.2 Pemilihan Jumper Kontrol Mode

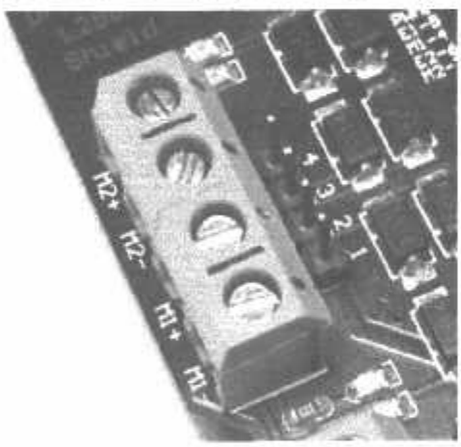
DfrDuino ini mendukung PWM dan Mode PLL (*Phased Locked Loop*) kontrol. Modus PWM menggunakan E1 dan E2 untuk menghasilkan sinyal PWM. Modus PLL menggunakan M1 dan M2 untuk menghasilkan sinyal kontrol fase, terlihat pada gambar 2.10.



Gambar 2.12 Pengaturan Kontrol Mode

2.4.3.3 Termial Motor

Dua motor DC yang terhubung ke terminal motor biru. Header male di belakang terminal yang sama dengan terminal motor yang terlihat pada gambar 2.11.



Gambar 2.13 Terminal Motor

2.4.3.4 Tabel Kebenaran Sinyal Kontrol

Pada sinyal kontrol terdapat logika yang dapat kita gunakan untuk menentukan masukan dan keluaran motor driver. Berikut adalah table kebenaran sinyal kontrol dari DfrDuino.

Tabel 2.4 Tabel Kebenaran

E1	M1	E2	M2		Keadaan
L	X	Motor 1 Disabled	L	X	Motor Maju
H	H	Motor 1 Backward	H	H	Motor Mundur
PWM	X	PWM Speed control	PWM	X	PWM Speed control

Keterangan : H adalah High level; L adalah Low level; PWM adalah Pulse Width Modulation signal; X adalah voltage level

2.4.3.5 Alokasi Pin

Pada driver motor DfrDuino terdapat pin yang dapat dikonfigurasikan sesuai keinginan. Pada setiap keluaran motor terdapat 2 pin yang berfungsi sebagai pengontrolan PWM dan pengontrolan arah motor (directional). Berikut ini adalah table alokasi pin pada DfrDuino.

Table 2.5 Alokasi Pin

"PWM Mode"	
Pin	**Function**
Digital 4	Motor 2 Direction control
Digital 5	Motor 2 PWM control
Digital 6	Motor 1 PWM control
Digital 7	Motor 1 Direction control

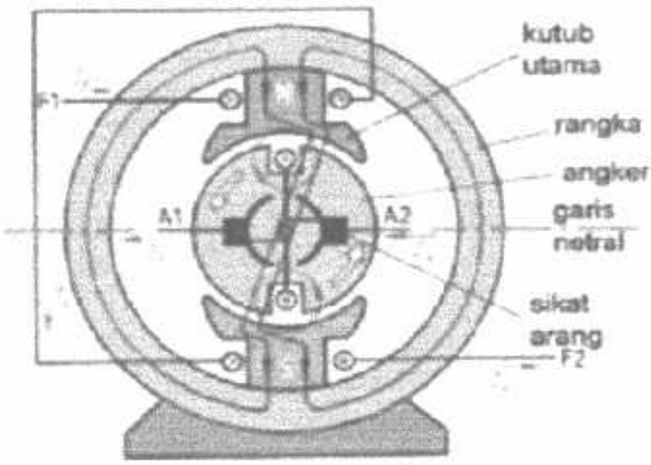
2.4.4 Motor DC

Motor DC merupakan jenis motor yang menggunakan tegangan searah sebagai sumber tenaganya. Dengan memberikan beda tegangan pada kedua terminal tersebut, motor akan berputar pada satu arah, dan bila polaritas dari tegangan tersebut dibalik maka arah putaran motor akan terbalik pula. Polaritas dari tegangan yang diberikan pada dua terminal menentukan arah putaran motor sedangkan besar dari beda tegangan pada kedua terminal menentukan kecepatan motor.

Motor DC memiliki 2 bagian dasar :

1) Bagian yang tetap/stasioner yang disebut stator. Stator ini menghasilkan medan magnet, baik yang dibangkitkan dari sebuah koil (elektro magnet) ataupun magnet permanen.
2) Bagian yang berputar disebut rotor. Rotor ini berupa sebuah koil dimana arus listrik mengalir.

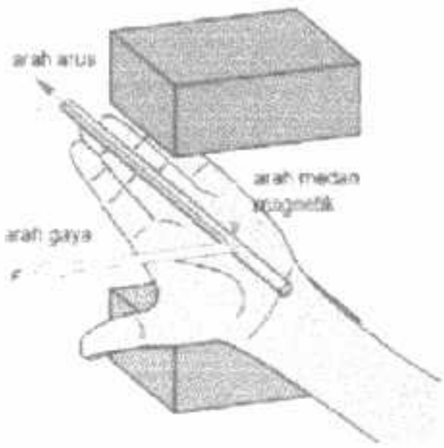
Gaya elektromagnet pada motor DC timbul saat ada arus yang mengalir pada penghantar yang berada dalam medan magnet. Medan magnet itu sendiri ditimbulkan oleh megnet permanen. Garis-garis gaya magnet mengalir diantara dua kutub magnet dari kutub utara ke kutub selatan. Menurut hukum gaya Lourentz, arus yang mengalir pada penghantar yang terletak dalam medan magnet akan menimbulkan gaya. Gaya F, timbul tergantung pada arah arus I, dan arah medan magnet B.



Ganbar 2.14 Konstruksi Motor DC

Belitan stator merupakan elektromagnet, dengan penguat magnet terpisah F1-F2. Belitan jangkar ditopang oleh poros dengan ujung-ujungnya terhubung ke komutator dan sikat arang A1-A2. Arus listrik DC pada penguat magnet mengalir dari F1 menuju F2 menghasilkan medan magnet yang memotong belitan jangkar. Belitan jangkar diberikan listrik DC dari A2 menuju ke A1. Sesuai kaidah tangan kiri jangkar akan berputar berlawanan jarum jam.

Gaya elektromagnet pada motor DC timbul saat ada arus yang mengalir pada penghantar yang berada dalam medan magnet. Medan magnet itu sendiri ditimbulkan oleh megnet permanen. Garis-garis gaya magnet mengalir diantara dua kutub magnet dari kutub utara ke kutub selatan. Menurut hukum gaya Lourentz, arus yang mengalir pada penghantar yang terletak dalam medan magnet akan menimbulkan gaya. Gaya F, timbul tergantung pada arah arus I, dan arah medan magnet B. Arah gaya F dapat ditentukan dengan aturan tangan kiri seperti pada gambar berikut.



Gambar 2.15 Penentuan Arah Gaya Pada Kawat Berarus Listrik Dalam Medan Magnet

BAB III

ANALISA DAN PERANCANGAN SISTEM

3.1 Analisa Sistem

Analisa ditunjukkan untuk memberikan gambaran secara umum tentang aplikasi dan memberikan solusi terhadap permasalahan yang dihadapi. Dalam analisa masalah dan penyelesaian dari masalah yang dihadapi.

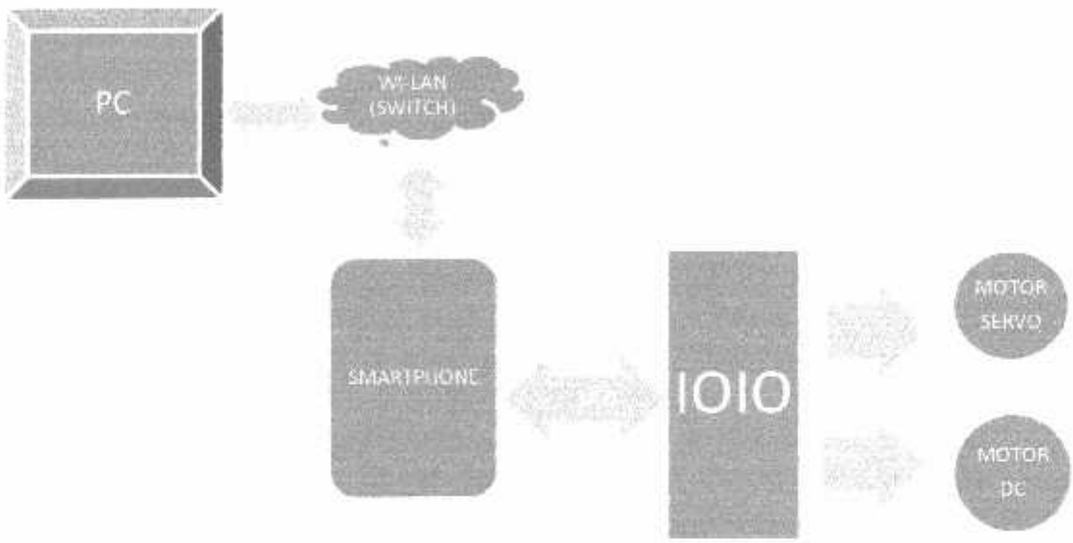
3.1.1 Identifikasi Masalah

Tahap identifikasi masalah merupakan tahapan paling awal untuk melakukan perancangan dan pembuatan aplikasi. Tahapan ini di gunakan untuk melakukan penelusuran permasalahan atau observasi untuk mendapatkan permasalahan umum dari permasalahan yang dihadapi. Di dalam tahapan ini juga dilakukan perumusan permasalahan yaitu menetapkan atau merumuskan permasalahan yang dihadapi, sehingga lebih fokus untuk mencari dan memecahkan permasalahan yang ada. Pada skripsi ini permasalahan yang dihadapi ada pada alat pengontrol robot kamera dapat saling berkomunikasi dengan bantuan wireless.

3.2 Gambaran Umum Sistem

Robot kamera ini terdiri dari PC , smartphone, modul IOIO, motor servo, dan motor DC. PC sebagai server akan terlebih dahulu membuat jaringan UDP kemudian menunggu data dari smartphone, Kemudian smartphone akan menginputkan IP Address PC untuk mengirimkan data. Data yang dikirim oleh smartphone adalah data kamera dan data pwm IOIO. Ketika PC telah menerima semua data tersebut maka semua fitur pada aplikas akan aktif secara otomatis dan selanjutnya PC akan mengirimkan lagi data untuk memerintahkan agar smartphone mengirim data melalui usb agar IOIO menjalankan motor servo dan motor DC.

Berikut desain gambaran umum sistem pengontrol robot menggunakan Smartphone Android yang ditunjukkan pada gambar 3.1 berikut ini.



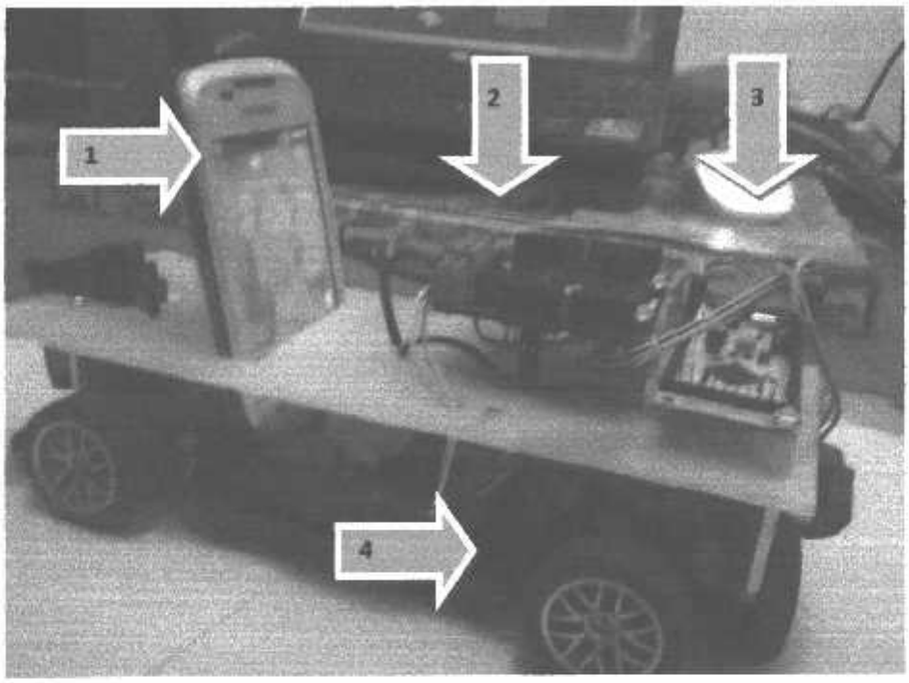
Gambar 3.1 Diagram Desain Sistem

Dari gambar 3.1 dapat dijelaaskan secara umum dan fungsi dari masing-masing blok sebagai berikut :

1. **Smartphone Android** berfungsi sebagai pengontrol robot yang sudah mempunyai kamera, wifi serta port usb yang sudah dibutuhkan dalam pengontol robot.
2. **IOIO** berfungsi untuk menghitung dan mengkonversi nilai dari sensor menjadi nilai sebuah sudut, kemudian menentukan arah robot.
3. **Wifi** berfungsi untuk mengirimkan data pergerakan dari PC dari Smartphone Android ke IOIO yang sudah terpasang pin.
4. **USB** berfungsi menghubungkan Smartphone Android dengan IOIO untuk mengirimkan data pwm.
5. **Motor Servo** berfungsi sebagai mekanik robot, untuk menggerakan kiri dan kanan.
6. **Motor DC** berfungsi sebagai mekanik robot untuk menggerakkan maju dan mundur.

3.3 Perancangan Perangkat Keras Robot

Pada gambar 3.2 adalah desain robot kamera secara keseluruhan yang digunakan pada rancang bangun skripsi ini.



Gambar 3.2 Design Robot Kamera

Keterangan gambar :

1. Smarphone Android
2. Dfrduino (driver motor DC)
3. IOIO
4. Motor DC

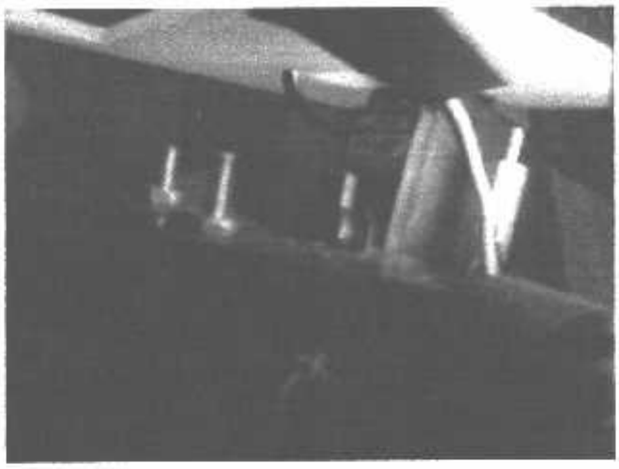
3.3.1 Desain Perangkat Keras

Pada desain perancangan perangkat keras akan dibahas penempatan seluruh komponen hardware pada robot. Pada desain ini bagaimana agar robot dapat berjalan selayaknya robot beroda, tanpa kelebihan beban dan supplai daya yang cukup.

3.3.1.1 Motor Servo

Pada perancangan robot kamera ini memakai RC Car yang telah dimodifikasi. Pada bagian depan robot memakai motor servo untuk menggerakkan robot ke arah kanan dan kiri

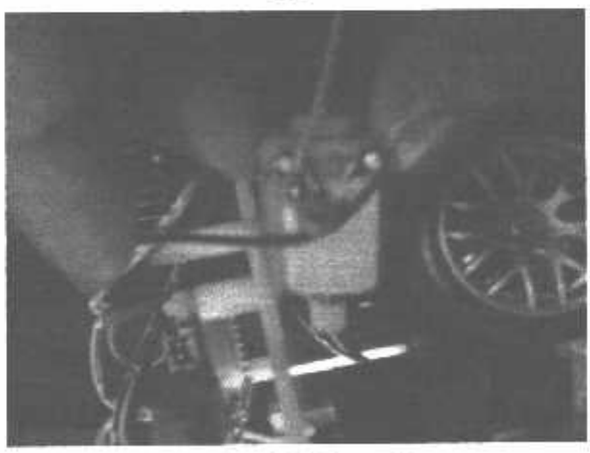
seperti pada gambar 3.3. Dengan menggunakan motor servo diharapkan pergerakan robot untuk maju dan mundur tetap ditengah, selain itu diharapkan dengan memakai motor servo pergerakan belok kanan dan belok kiri dapat direspon dengan cepat oleh motor servo , seperti halnya motor RC berukuran besar pada umumnya.



Gambar 3.3 Motor Servo

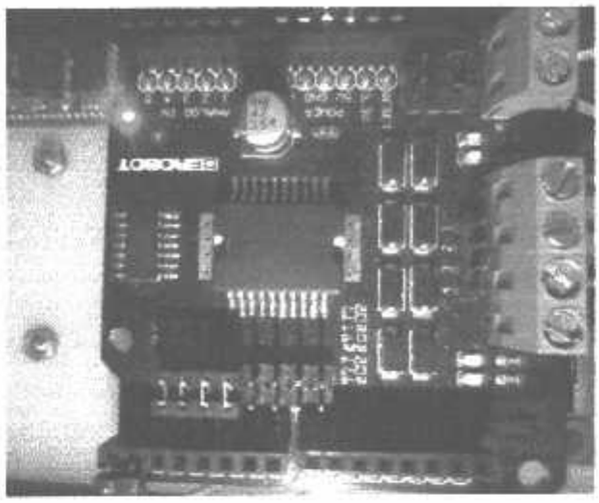
3.3.1.2 Motor Dc dan DfrDuino

Pada robot kamera ini , gerakan maju mundur diatur oleh motor Dc. Motor Dc membutuhkan driver untuk dibaca oleh IOIO, pada robot ini memakai Dfrduino sebagai motor driver. Dfrduino memiliki 2 inputan, yang pertama ialah input pwm dan yang kedua adalah direction untuk memutar balikkan arah motor dc. Pada gambar 3.4 terlihat motor Dc yang digunakan robot kamera ini untuk menggerakkan 2 buah roda belakang .



Gambar 3.4 Motor Dc

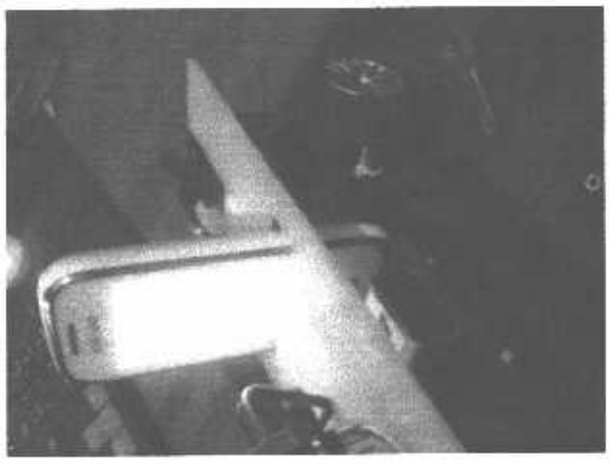
dan pada gambar 3.5 terlihat DfrArduino yang dipasang diatas acrylic



Gambar 3.5 Driver Motor Dfrduino

3.3.1.3 Penyangga Smartphone

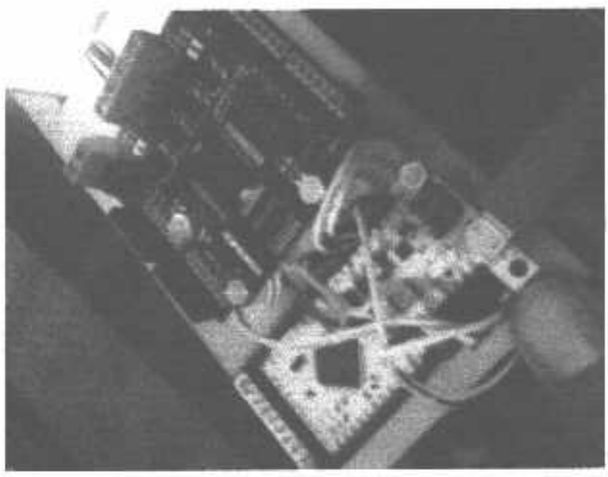
Pada robot ini terdapat penyangga Smartphone yang dibuat dengan cara melubangi acrylic dengan kedalaman 3 cm dan lebar 6 cm, kedalaman diukur agar jangan sampai menutupi layar Smartphone. Penyangga ini ditempatkan ke posisi tengah, lalu dikondisikan posisi kamera Smartphone dapat melihat area di sekitarnya. Dengan penyangga ini diharapkan pada saat robot bergerak Smartphone tidak goyang apalagi sampai terjatuh seperti yang terlihat pada gambar 3.6



Gambar 3.6 Penyangga Robot

3.3.1.4 IOIO

Pada robot ini IOIO bekerja sebagai mikrokontroler, yang berperan sebagai penggerak motor servo dan motor Dc. IOIO dipasang di paling belakang robot seperti pada gambar 3.7.



Gambar 3.7 IOIO

3.3.2 Catu Daya

Pada robot ini memakai 2 catu daya, yang pertama ialah catu daya 6 V yang dipakai untuk mensupplai IOIO dan motor servo, terlihat pada gambar 3.8 catu daya ini berisi 4 baterai 1,5V yang diseri dan tempat yang digunakan adalah bawaan tempat baterai RC car.

Gambar 3.8 Catu Daya 6V

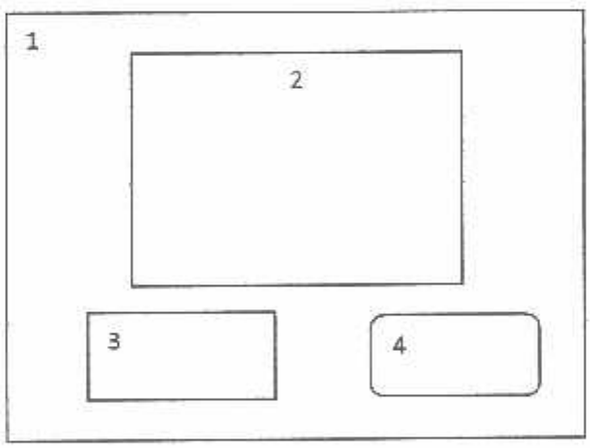
Lalu baterai 9V dipakai untuk mensuplai motor Dc dan driver motoryang dapat dilihat pada gambar 3.9

Gambar 3.9 Catu daya 9V

3.4 Perencanaan Perangkat Lunak

3.4.1 Perencanaan Aplikasi PC

Pada perencanaan aplikasi PC dibutuhkan tampilan video streaming kamera dan accelerometer untuk monitoring dan juga tombol arah sebagai pengontrol robot. Berikut ini adalah perancangan dari aplikasi PC



Gambar 3.10 Design Aplikasi PC

Keterangan Gambar :

1. Background

   Pada background akan di atur menyesuaikan layar komputer.

2. Tampilan Kamera.

Pada layout 2 akan menampilkan kamera dari smartphone dengan resolusi 320 x 480.

3. Tombol Arah

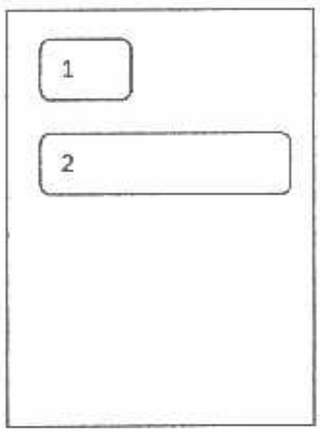
   Tombol arah akan menginstruksikan robot untuk maju, mundur ,kiri dan kanan.

4. Data Accelerometer

   Pada layout 4 akan ditampilkan data accelometer pada Smartphone.

3.4.1 Perencanaan Aplikasi Smartphone

Pada Aplikasi Smartphone terdapat tombol on/off untuk mengaktifkan koneksi router dan sebuah kotak dialog untuk mengisi IP Address PC. Berikut ini adalah perancangan dari aplikasi Smartphone.



Gambar 3.11 Design Antarmuka Smartphone

Keterangan Gambar :

1. Tombol Start

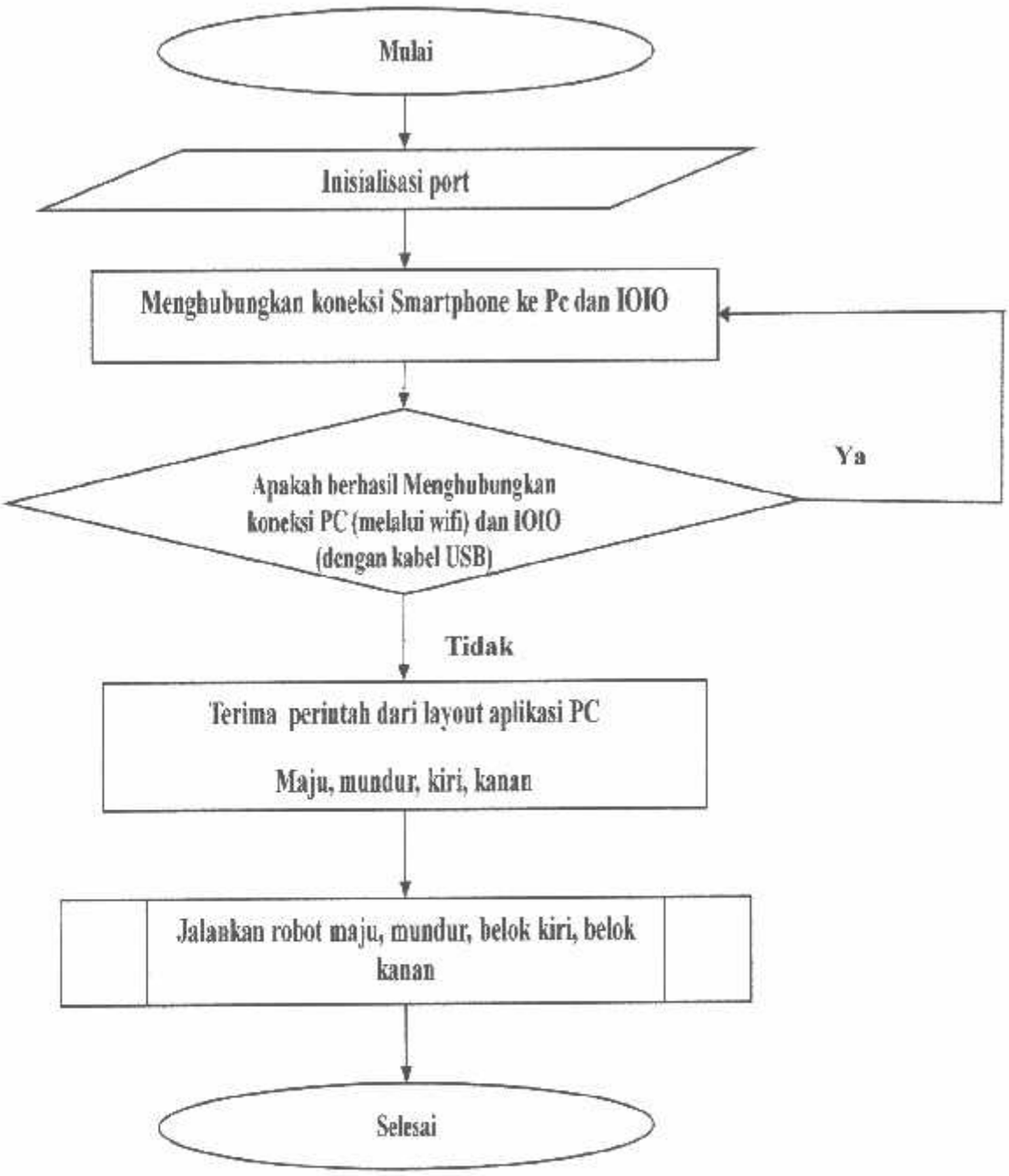
   Pada tombol start terdapat pilihan ON/OFF , ketika ON maka smartphone akan mengirimkan data kepada konputer

2. IP address

   Pada kolom ini, terdapat masukkan angka yang berisi IP Adrress PC yang berisi maksimal 9 angka.

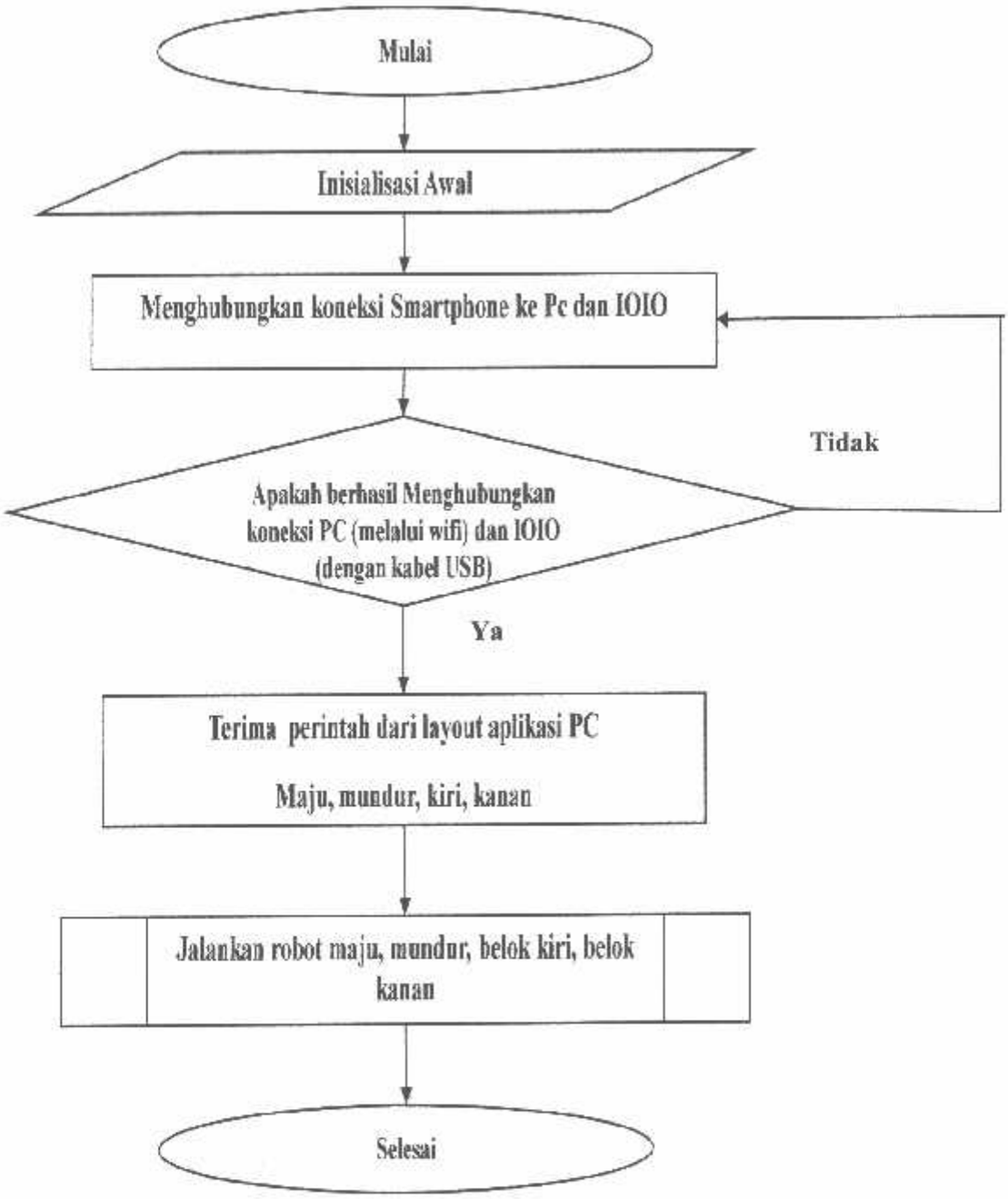
3.5 Flowchart

3.5.1 Flowchart Aplikasi Smartphone



Gambar 3.12 Flowchart Aplikasi Smartphone

3.5.2 Flowchart Aplikasi PC



Gambar 3.13 Flowchart Aplikasi PC


INSTITUT TEKNOLOGI NASIONAL
ITN
MALANG

BAB IV
IMPLEMENTASI DAN PENGUJIAN SISTEM

Tahap pengujian sistem merupakan proses menjalankan aplikasi yang telah dirancang yang dapat dijalankan. Tahapan ini merupakan lanjutan dari proses perancangan sesuai dengan spesifikasi dan desain sistem. Untuk mengetahui sistem yang dirancang sesuai dengan fungsi yang diharapkan, dilakukan pengujian terhadap sistem aplikasi tersebut baik secara keseluruhan atau subsistem. Selain itu pengujian juga dilakukan agar dapat menemukan beberapa permasalahan yang mungkin timbul pada saat alat ini beroperasi untuk kemudian diperbaiki sampai pada tingkat kesalahan sekecil mungkin sehingga didapatkan hasil yang baik. Berikut penjelasan mengenai prosedur pengontro robot.

4.1 Pengujian Aplikasi Smartphone

4.1.1 Pengujian Koneksi Smartphone dan IOIO

Pengujian pada aplikasi Smartphone dilakukan dengan menjalankan aplikasi yang telah dibuat dengan mencoba koneksi IOIO . Hal pertama yang dilakukan adalah mengaktifkan USB Debugging pada Smartphone agar IOIO dapat terhubung seperti gambar 4.1, dengan cara Aplication setting > Development.

Gambar 4.1 Tampilan USB Debugging

Jika telah terhubung maka ioio akan mengcharge dan juga terdapat label USB seperti pada gambar 4.1. Jika telah terhubung maka hubungkan power ke ioio maka led bewarna merah

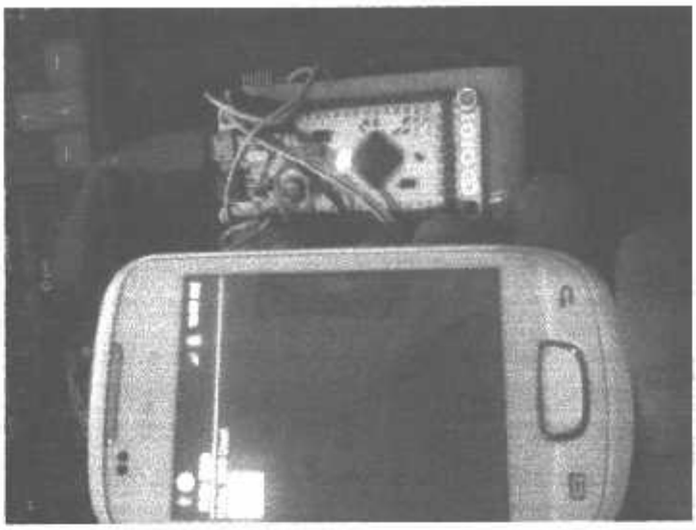
akan menyala lalu coba jalankan aplikasi hello ioio untuk meyalakan led bewarna kuning seperti gambar 4.2

```
package ioio.examples.hello;

public class MainActivity extends AbstractIOIOActivity {
 private ToggleButton button_;

 public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);
        button_ = (ToggleButton) findViewById(R.id.button);
 }
 class IOIOThread extends AbstractIOIOActivity.IOIOThread {
                        private DigitalOutput led_;

        @Override
        protected void setup() throws ConnectionLostException {
                led_ = ioio_.openDigitalOutput(0, true);
        }
        protected void loop() throws ConnectionLostException {
                led_.write(!button_.isChecked());
                try {
                        sleep(100);
                } catch (InterruptedException e) {
                }
        }
 }
 protected AbstractIOIOActivity.IOIOThread createIOIOThread() {
        return new IOIOThread();
}
```



Gambar 4.2 Koneksi Antara Smartphone dan IOIO

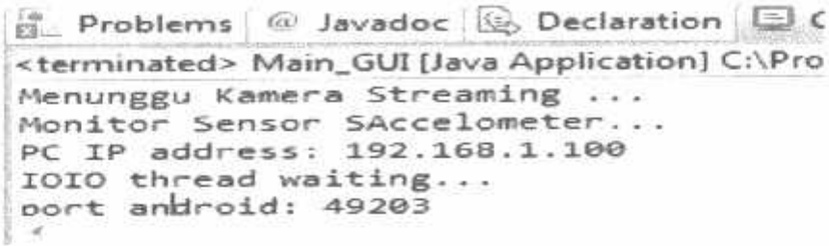
4.1.2 Pengujian Aplikasi Smartphone ke PC

Pada pengujian ini, kita jalankan aplikasi yang telah dibuat. Pada aplikasi ini terdapat satu button On/Off dan satu kotak dialog. Button berfungsi menghubungkan Smartphone pada PC dan IOIO, sedangkan kotak dialog berfungsi sebagai masukkan *IP Address* PC. Pada pengujian ini *IP Address* Pc 192.168.1.100, ketika kita menekan button maka Smartphone akan mencoba terhubung dengan Pc dan menunggu perintah. Jika berhasil terhubung maka akan tampil dialog " start streaming " seperti pada gambar 4.3

Gambar 4.3 Pengujian Aplikasi Smartphone

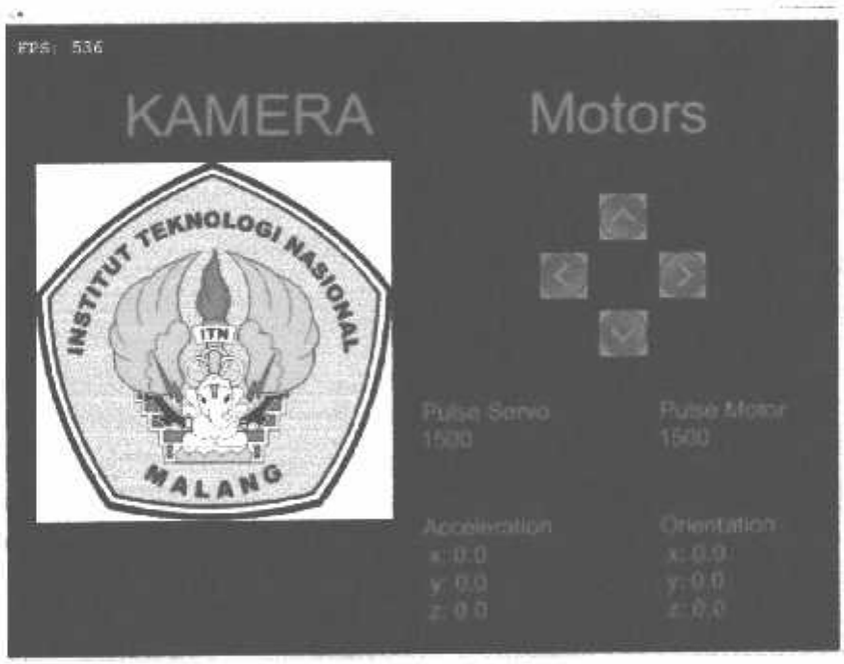
4.2 Pengujian Aplikasi PC

Pengujian aplikasi pada PC dilakukan dengan menjalankan aplikasi pada eclipse yang sudah terinstal pada PC. Ketika aplikasi ini dijalankan, aplikasi akan menunggu data yang dikirim oleh Smartphone akan tampil seperti pada gambar 4.4.



Gambar 4.4 Tampilan Running Eclipse

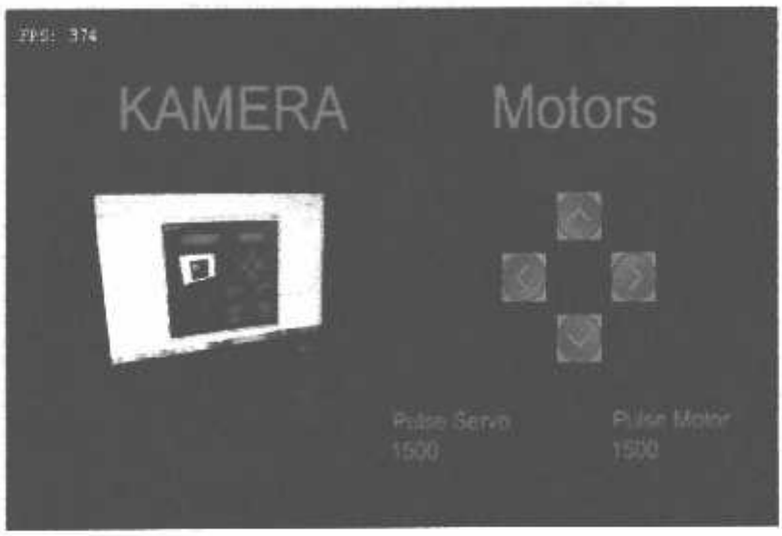
Setelah kita menjalankan aplikasi ini maka tampilan awal yang akan ditampilkan seperti pada gambar 4.5



Gambar 4.5 Tampilan Awal Aplikasi Pc

Ketika mendapatkan data dari smartphone otomatis smartphone akan mengirimkan data kamera secara streaming lalu aplikasi akan mengaktifkan fungsi dari tombol arah maju, mundur, kanan, kiri. Di waktu yang bersamaan smartphone telah siap menerima perintah dari PC untuk menjalankan robot. Pada gambar 4.5 dapat dilihat koneksi antara Smartphone dan PC telah berhasil. Pada aplikasi ini penulis menambahkan fungsi

accelometer untuk monitor tambahan apakah Smartphone dan Pc telah terhubung dengan baik, karena parameter x,y dan z pada accelometer android sudah cukup akurat dan sering digunakan para pengembang untuk mengendalikan robot atau perangkat lainnya.



Gambar 4.6 Tampilan Aplikasi PC Setelah Dijalankan

4.3 Pengujian Alat

Pengujian alat bertujuan untuk mengetahui kinerja sistem secara keseluruhan,apakah sudah sesuai dengan spesifikasi yang direncanakan.

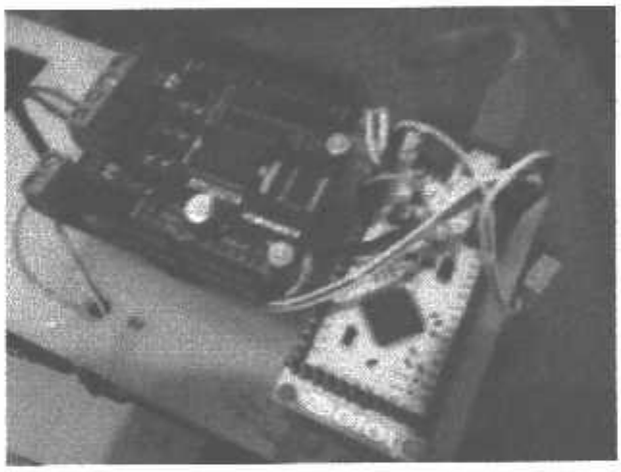
1. Langkah pertama adalah menyalakan alat dengan menghubungkan 2 baterai dan kabel usb ke smartphone. Pastikan Smartphone dan PC telah terhubung wireless dalam satu jaringan.
2. Setelah lampu led indicator di IOIO menyala dan Smartphone Android akan terhubung sambil mencharge.
3. Jalankan aplikasi pada pc sampai muncul tulisan " menunggu streaming kamera, menunggu koneksi IOIO dan PC address".
4. Setelah itu jalankan aplikasi smartphone dan tombol start.
5. PC akan otomatis menerima streaming video dari smartphone jika koneksi berhasil
6. Setelah itu tombol pada aplikasi PC telah aktif dan siap untuk menjalankan robot

Pada pengujian alat kita akan mengkonfigurasikan pin yang dipakai dalam menjalankan alat. Pada table 4.1 akan dijelaskan pin – pin IOIO yang dipakai.

Table 4.1 Konfigurasi Pin yang Dipakai

Pin IOIO	Keterangan
Vin.	Sebagai sumber baterai
Pin 1	Sebagai groung
Pin 3	Sebagai direction motor dc
Pin 4	Sebagai Pwm motor dc
Pin 6	Sebagai Pwm motor servo
3.3v	Sebagai + servo
GND 1	Sebagai - servo
GND2 dan 5V	Sebagai penghubung 2 baterai

Pada gambar 4.7 menunjukan semua pin yang dibutuhkan telah tersambung dan robot siap dijalankan.



Gambar 4.7 koneksi pin pada IOIO

4.3.1 Pengujian arah gerakan robot beroda

Untuk mengetahui arah gerakan robot beroda dengan cara melihat nilai pwm pada tampilan aplikasi PC (Gambar 4.4). Pada source code robot ini, motor dc ditetapkan nilainya agar robot ini berjalan dengan kecepatan yang sama, sedangkan motor servo nilainya berubah – ubah sesuai derajat derajat arah perputarannya. Gerakan robot beroda dapat dilihat di Tabel 4.2.

Table 4.2 Pergerakan Robot

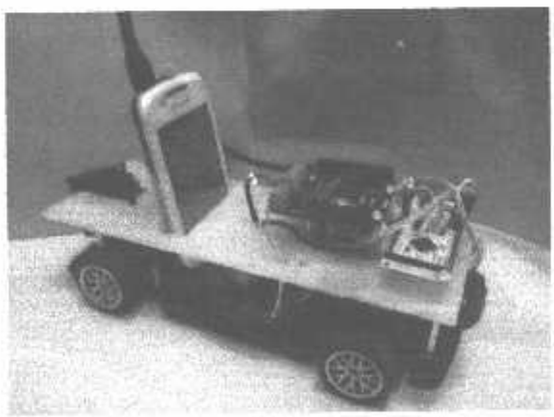
Nilai sensor Accelometer	Arah gerakan
Motor bernilai 1900	Robot beroda bergerak maju
Motor bernilai 1400	Robot beroda bergerak mundur
Servo bernilai 1350	Robot beroda bergerak kiri
Servo bernilai 1775	Robot beroda bergerak kanan

4.3.2 Pengujian Jarak Jangkauan Robot

Untuk mengetahui jarak jangkauan komunikasi robot dilakukan percobaan dengan menjalankan robot menjauh dari client. Pada percobaan ini, robot berjalan hingga kejauhan 11 meter yang dapat dilihat pada table 4.3.

Table 4.3 Jarak Jangkauan Robot

Jarak	**Komukasi Robot**
1 meter	Berjalan baik
3 meter	Berjalan baik
5 meter	Berjalan baik
7 meter	Berjalan baik
9 meter	Tidak berjalan baik
11 meter	Tidak berjalan baik



Gambar 4.8 Robot Kamera

BAB V
PENUTUP

5.1 Kesimpulan

Setelah melakukan pembuatan dan perencanaan sistem kemudian dilakukan pengujian dan analisa, maka kesimpulannya yaitu sebagai berikut:

1. Sistem ini telah berhasil melakukan koneksi antara Smartphone Android dengan robot beroda melalui media Wifi dengan platform yang berbeda. Palatform yang digunakan adalah Java pada aplikasi PC, Android pada Smartphone dan IOIOLib Pada Mikrokontroler.
2. Sistem ini telah berhasil melakukan pengendalian robot beroda dengan Smartphone sebagai otaknya ,yang beroperasi pada sistem operasi Android dan memanfaatkan kamera smartphone secara langsung.
3. Jalur komunikasi antara Wifi pada Smartphone Android dan Wifi memungkinkan pengontrolan robot dengan jarak jauh.
4. Berdasarkan hasil pengujian jarak optimal komunikasi jarak jauh (wireless) robot ini berjarak 7 meter.

5.2 Saran

Untuk pengembangan sistem ini, penulis memberikan saran yang dapat dilakukan,beberapa saran tersebut antara lain :

1. Untuk mengontrol pergerakan robot beroda, disarankan minimal menggunakan firmware Android versi 2.3 Gingerbread atau firmware yang sudah di modifikasi atau custom ROM.
2. Untuk pengembangan selanjutnya, komunikasi WiFi antara robot beroda dengan Smartphone Android dianjurkan menggunakan metode TCP/IP yang memungkinkan Client-Server dapat diatur pada Smartphone.
3. Pada sitem komunikasi UDP tidak memiliki recovery data karena bersifat conectionless, diharapkan pada pengembangan selanjutnya memakai TCP/IP yang memiliki recovery paket data agar koneksi tidak terputus ketika terjadi gangguan.
4. Dianjurkan memakai Smartphone dengan spesifikasi lebih tinggi, karena Pada alat ini menggunakan Samsung Mini yang spesikasinya hanya resolusi kamera 3.2 dan prosesor 600mhz terjadi hang, blackscreen ketika terhubung dengan IOIO.

DAFTAR PUSTAKA

A. Gow, Gordon & K.Smith Richard. *Mobile and Wireless Communication : An Introduction.* Poland 2006

Borko Furht, Ph.D & Mohammad Ilyas, Ph.D.*Wireless Internet Handbook Technologies Standards and Applications, penerbit CRC Press,* Washington D.C 2013

Jan Graba. *An Introduction to Network Programming with Java*, Penerbit Springer, USA, 2013

Kimmo & Karvinen, Tero. *Make arduino bots and gadgets*, Penerbit O'REILLY, Canada 2011

Paul Deitel, Harvey Deitel, Abbey Deitel. *Android, How to Program,* Penerbit Deitel & Associates Inc, San Francisco 2013

Reza N. lazar. *Theory of Applied Robotics Kinematics, Dynamics, and Control,* Penerbit Springer, USA 2010

Safaat, Nazaruddin H. *Pemrograman aplikasi mobile smartphone dan tablet Pc berbasis android,* Penerbit Informatika, Bandung 2012

Simon Monk. Making Android Accessories with IOIO, Penerbit O'REILLY, Canada 2012

http://akbarulhuda.wordpress.com/2010/04/01/mengenal-motor-servo Mengenal motor servo (diakses tanggal 15-12-2013)

http://hellcode.wordpress.com *android and arduino bluetooth and wifi communication* (diakses tanggal 05-09-2012)

http://blog.uad.ac.id/bambangis/2011/04/01/definisi-udp-user-datagram-protocol Konsep UDP (diakses tanggal 10-10-2012)

https://github.com/ytai/ioio/wiki/IOIOdigital-IO *IOIO digital input/ output*(diakses tanggal 20-08-2012)

http://id.wikipedia.org/wiki/Android_%28sistem_operasi%29 Pengertian Sistem Operasi Android (diakses tanggal 20-09-2012)

https://play.google.com/store/apps/details *IOIO Bot communication*(diakses tanggal 15-12-2013)

http://www.arduino.cc How to set up your motor driver(diakses tanggal 15-09-2012)

http://www.gsmarena.com/samsung_galaxy_mini_s5570-3725.php Spesifikasi Samsung Galaxy Mini (diakses tanggal 4-12-2013)

LAMPIRAN

PERKUMPULAN PENGELOLA PENDIDIKAN UMUM DAN TEKNOLOGI NASIONAL MALANG
INSTITUT TEKNOLOGI NASIONAL MALANG
FAKULTAS TEKNOLOGI INDUSTRI
FAKULTAS TEKNIK SIPIL DAN PERENCANAAN
PROGRAM PASCASARJANA MAGISTER TEKNIK
PT. BNI (PERSERO) MALANG
BANK NIAGA MALANG
Kampus I : Jl. Bendungan Sigura-gura No. 2 Telp. (0341) 551431 (Hunting), Fax. (0341) 553015 Malang 65145
Kampus II : Jl. Raya Karanglo, Km 2 Telp. (0341) 417636 Fax. (0341) 417634 Malang

BERITA ACARA UJIAN SKRIPSI
FAKULTAS TEKNOLOGI INDUSTRI

Nama	:	**Iwan Pernando P**
Nim	:	**08.12.516**
Jurusan	:	**Teknik Elektro**
Konsentrasi	:	**Teknik Komputer S-1**
Masa Bimbingan	:	**Semester Ganjil 2013-2014**
Judul	:	**RANCANG BANGUN PENGONTROLAN ROBOT KAMERA BERBASIS ANDROID**

Dipertahankan dihadapan Tim Penguji Skripsi Jenjang Program Strata Satu (S-1)

Pada Hari : Selasa
Tanggal : 18 Februari 2014
Dengan Nilai : 80,95 (A)

Panitia Ujian Skripsi :

Ketua Majelis Penguji

M. Ibrahim Ashari, ST, MT
NIP.P. 1030100358

Sekretaris Majelis Penguji

Dr. Eng. Aryuanto Soetedjo, ST, MT
NIP.Y.1030800417

Anggota Penguji :

Dosen Penguji I

M. Ibrahim Ashari, ST, MT
NIP.P. 1030100358

Dosen Penguji II

Yuli Wahyuni, ST, MT
NIP. P.1031200456

INSTITUT TEKNOLOGI NASIONAL MALANG
FAKULTAS TEKNOLOGI INDUSTRI
JURUSAN TEKNIK ELEKTRO S-1
Jl. Karanglo KM.2 Malang

FORMULIR PERBAIKAN SKRIPSI

Dalam pelaksanaan Ujian Skripsi Jenjang Strata 1 Jurusan Teknik Elektro Konsentrasi Teknik Komputer, maka perlu adanya perbaikan skripsi untuk mahasiswa :

Nama	: **IWAN PERNANDO PANJAITAN**
Nim	: **08.12.516**
Jurusan	: **Teknik Elektro S-1**
Konsentrasi	: **Teknik Komputer**
Masa Bimbingan	: **Semester Ganjil 2013-2014**
Judul Skripsi	: **RANCANG BANGUN PENGONTROLAN ROBOT KAMERA BERBASIS ANDROID**

No	Penguji	Tanggal	Uraian	Paraf
1.	Penguji I	18 Februari 2014	1. Daftar table dan daftar gambar 2. Gambar 3. Kesimpulan	
2	Penguji II	18 Februari 2014	1. Pengujian jarak jangkauan 2. Kesimpulan ditambahkan berdasarkan hasil 3. Daftar pustaka	

Disetujui:

Penguji I

M. Ibrahim Ashari, ST, MT
NIP.P.1030100358

Penguji II

Yuli Wahyuni, ST, MT
NIP. P.1031200456

Mengetahui:

Dosen Pembimbing I

Ir. Yusuf Ismail Nakhoda, MT
NIP.Y.1018800189

Dosen Pembimbing II

Dr. Eng. Aryuanto Soetedjo, ST,MT
NIP.Y.1030800417

INSTITUT TEKNOLOGI NASIONAL MALANG
FAKULTAS TEKNOLOGI INDUSTRI
JURUSAN TEKNIK ELEKTRO

Formulir Perbaikan Ujian Skripsi

Dalam pelaksanaan Ujian Skripsi Janjang Strata 1 Jurusan Teknik Elektro Konsentrasi T. Energi Listrik / T. Elektronika / T.Infokom, maka perlu adanya perbaikan skripsi untuk mahasiswa :

NAMA : Iwan Fernando . P.
NIM : 08.13.516
Perbaikan meliputi :

1) pengujian jarak jangkauan.

2) kesimpulan ditambahkan berdasarkan hasil pengujian

3) Daftar Pustaka

Malang, 18 Februari 2014

(Yuli Wahyuni, ST., MT)

LAMPIRAN

Script Android (smartphone) dan IOIO

package IOIO_car;

```
import android.app.Activity;
import android.hardware.SensorManager;
import android.os.Bundle;
import android.view.SurfaceView;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.EditText;
import android.widget.Toast;
import android.widget.ToggleButton;


public class Android_Activity extends Activity
{
    Main_thread simulator;
    ToggleButton togglebutton;
    EditText ip_text;
  SensorManager sm = null;
  SurfaceView view;
  Sensors_thread the_sensors=null;
  String IP_address;
    Android_Activity the_app;

    @Override
    protected void onCreate(Bundle savedInstanceState)
    {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);
        view = new SurfaceView(this);
     sm = (SensorManager) getSystemService(SENSOR_SERVICE);
        ip_text = (EditText) findViewById(R.id.IP_edit_txt);
        togglebutton = (ToggleButton) findViewById(R.id.CameraButton);
        togglebutton.setOnClickListener(new btn_listener());
        the_app = this;
    }

    @Override
    protected void onResume()
    {
        super.onResume();
    }

    protected void onStop()
    {
        super.onStop();
        simulator.stop_simu();
        this.finish();
    }

    private class btn_listener implements OnClickListener
```

```
    {
        public void onClick(View v)
        {
            // Perform action on clicks
            if (togglebutton.isChecked())
            {
                IP_address = ip_text.getText().toString();

                simulator = new Main_thread(the_app, view, sm, IP_address);
                the_sensors = simulator.the_sensors;
                sm.registerListener(the_sensors,
                        SensorManager.SENSOR_ORIENTATION
|SensorManager.SENSOR_ACCELEROMETER,
                        SensorManager.SENSOR_DELAY_UI);

                simulator.start();

                Toast.makeText(Android_Activity.this, "Start streaming",
Toast.LENGTH_SHORT).show();
            } else
            {
                simulator.stop_simu();
                sm.unregisterListener(the_sensors);
                Toast.makeText(Android_Activity.this, "Stop streaming",
Toast.LENGTH_SHORT).show();
            }
        }
    }
}
```

```
package IOIO_car;

import java.io.ByteArrayOutputStream;
import java.net.DatagramPacket;
import java.net.DatagramSocket;
import java.net.InetAddress;
import android.graphics.Bitmap;
import android.hardware.Camera;
import android.hardware.Camera.PreviewCallback;
import android.util.Log;
import android.view.SurfaceView;

public class Cam_thread
{
	Camera mCamera;

	public static int HEADER_SIZE = 5;
	public static int DATAGRAM_MAX_SIZE = 1450 - HEADER_SIZE;
	int frame_nb = 0;
	int size_packet_sent = 0;
	InetAddress serverAddr;
	DatagramSocket socket;

	Bitmap mBitmap;
	int[] mRGBData;
	int width_ima, height_ima;
	private static final String TAG = "IP_cam";

	SurfaceView parent_context;

	private boolean STOP_THREAD;
	String ip_address;

	public Cam_thread(SurfaceView context, String ip)
	{
		parent_context = context;
		ip_address = ip;
	}

	private void init()
	{
		try
		{
		serverAddr = InetAddress.getByName(ip_address);
			socket = new DatagramSocket();

			mCamera = Camera.open();
			Camera.Parameters parameters = mCamera.getParameters();
			parameters.setPreviewSize(320, 240);
			parameters.setPreviewFrameRate(30);
			parameters.setSceneMode(Camera.Parameters.SCENE_MODE_SPORTS);
			parameters.setFocusMode(Camera.Parameters.FOCUS_MODE_AUTO);
			parameters.setColorEffect(Camera.Parameters.EFFECT_NONE);
			mCamera.setParameters(parameters);
```

```
                mCamera.setPreviewDisplay(parent_context.getHolder());

                mCamera.setPreviewCallback(new cam_PreviewCallback());
                mCamera.startPreview();
            }
            catch (Exception exception)
            {
                Log.e(TAG, "Error: ", exception);
            }
        }

    public void start_thread()
    {
      init();
    }

    public void stop_thread()
    {
      STOP_THREAD = true;
            socket.close();
    }

      public void send_data_UDP()
      {
            if(mBitmap != null)
            {
                int size_p=0,i;
                ByteArrayOutputStream byteStream = new ByteArrayOutputStream();

                mBitmap.compress(Bitmap.CompressFormat.JPEG, 50, byteStream);
                // Play with compression rate to change overall size

                byte data[] = byteStream.toByteArray();
                Log.e(TAG, "SIZE: " + data.length);

                int nb_packets = (int) Math.ceil(data.length /
(float)DATAGRAM_MAX_SIZE);
                int size = DATAGRAM_MAX_SIZE;

                // Loop through slices of 1024...
                for(i = 0; i < nb_packets; i++)
                {
                    if(i >0 && i == nb_packets-1) size = data.length - i *
DATAGRAM_MAX_SIZE;

                    // Set additional header
                    byte[] data2 = new byte[HEADER_SIZE + size];
                    data2[0] = (byte)frame_nb;
                    data2[1] = (byte)nb_packets;
                    data2[2] = (byte)i;
                    data2[3] = (byte)(size >> 8);
                    data2[4] = (byte)size;

                    // Copy current slice to byte array
```

```
                        System.arraycopy(data, i * DATAGRAM_MAX_SIZE, data2,
HEADER_SIZE, size);

                        try
                        {
                                size_p = data2.length;
                                DatagramPacket packet = new DatagramPacket(data2,
size_p, serverAddr, 9000);
                                socket.send(packet);
                        } catch (Exception e) {   Log.e(TAG, "Error: ", e);}
                    }
                    frame_nb++;

                    if(frame_nb == 128) frame_nb=0;
                }
        }


        static public void decodeYUV420SP(int[] rgb, byte[] yuv420sp, int width, int
height)
        {
                final int frameSize = width * height;

                for (int j = 0, yp = 0; j < height; j++) {
                    int uvp = frameSize + (j >> 1) * width, u = 0, v = 0;
                    for (int i = 0; i < width; i++, yp++) {
                        int y = (0xff & ((int) yuv420sp[yp])) - 16;
                        if (y < 0) y = 0;
                        if ((i & 1) == 0) {
                            v = (0xff & yuv420sp[uvp++]) - 128;
                            u = (0xff & yuv420sp[uvp++]) - 128;
                        }

                        int y1192 = 1192 * y;
                        int r = (y1192 + 1634 * v);
                        int g = (y1192 - 833 * v - 400 * u);
                        int b = (y1192 + 2066 * u);

                        if (r < 0) r = 0; else if (r > 262143) r = 262143;
                        if (g < 0) g = 0; else if (g > 262143) g = 262143;
                        if (b < 0) b = 0; else if (b > 262143) b = 262143;

                        rgb[yp] = 0xff000000 | ((r << 6) & 0xff0000) | ((g >> 2) &
0xff00) | ((b >> 10) & 0xff);
                    }
                }
        }

        private class cam_PreviewCallback implements PreviewCallback
        {
                @Override
                public void onPreviewFrame(byte[] data, Camera camera)
                {
                    if(STOP_THREAD == true)
                    {
```

```
                    mCamera.setPreviewCallback(null);
                    mCamera.stopPreview();
                    mCamera.release();
                    mCamera = null;
                    return;
                }

                if (mBitmap == null)                                    {
                    Camera.Parameters params = camera.getParameters();
                    width_ima = params.getPreviewSize().width;
                    height_ima = params.getPreviewSize().height;

                    mBitmap = Bitmap.createBitmap(width_ima, height_ima,
Bitmap.Config.RGB_565);
                    mRGBData = new int[width_ima * height_ima];
                }

                decodeYUV420SP(mRGBData, data, width_ima, height_ima);
                mBitmap.setPixels(mRGBData, 0, width_ima, 0, 0, width_ima,
height_ima);

                send_data_UDP();
            }
        }

}
```

```
package IOIO_car;

import java.io.IOException;
import java.net.DatagramPacket;
import java.net.DatagramSocket;
import java.net.InetAddress;

import ioio.lib.api.DigitalOutput;
import ioio.lib.api.IOIO;
import ioio.lib.api.IOIOFactory;
import ioio.lib.api.PwmOutput;
import ioio.lib.api.exception.ConnectionLostException;
import android.util.Log;

public class IOIO_Thread extends Thread
{
	/** Subclasses should use this field for controlling the IOIO. */
	protected IOIO ioio_;
	private boolean abort_ = false;
	private PwmOutput servo, motor;
	int size_p;
	int servo_val = 1500, motor_val=1500;
	DigitalOutput D1A;

    final String tag = "Sensors";
	InetAddress serverAddr;
	DatagramSocket socket;
	String ip_address;

	Android_Activity main_app;
	boolean START = true;
	int a_nb=0;


	public IOIO_Thread(Android_Activity app, String ip)
	{
		main_app = app;
		ip_address = ip;
	try
	{
		serverAddr = InetAddress.getByName(ip_address);
		socket = new DatagramSocket();
	}
	catch (Exception exception)
	{
		Log.e(tag, "Error: ", exception);
	}
	}

	/** Not relevant to subclasses. */
	@Override
	public final void run()
	{
		super.run();
		while (true)
```

```
        {
                try
                {
                        synchronized (cmds)
                        {
                                if (abort_) {break;}
                                ioio_ = IOIOFactory.create();
                        }
                        ioio_.waitForConnect();
                        setup();
                        while (true)
                        {
                                loop();
                        }
                }
                catch (ConnectionLostException e)
                {
                        if (abort_) {break;}
                }
                catch (Exception e)
                {
                        Log.e("AbstractIOIOActivity","Unexpected exception
caught", e);
                        ioio_.disconnect();
                        break;
                }
                finally
                {
                        try {ioio_.waitForDisconnect();
                        }
                        catch (InterruptedException e) {}
                }
        }
    }

    protected void setup() throws ConnectionLostException
    {
        D1A = ioio_.openDigitalOutput(4, false);

        motor = ioio_.openPwmOutput(3, 1000);
        servo = ioio_.openPwmOutput(6, 50);

        servo.setPulseWidth(1500);
        motor.setPulseWidth(1500);

        //..................... send phone IP address to computer
.....................//
        byte[] data = new byte[1];
        data[0] = (byte) (1);
        size_p = data.length;
        DatagramPacket packet = new DatagramPacket(data, size_p, serverAddr,
9002);
        try
        {
```

```
                socket.send(packet);

            }
            catch (IOException e) {Log.e("IOIO_thread","error: ", e);}
        }

        protected void loop() throws ConnectionLostException
        {
            if (motor_val > 1500 )
            {
                DIA.write(false);
            }
            else
            {
                DIA.write(true);
            }

            try
            {
                byte[] data2 = new byte[4];
                DatagramPacket receivePacket = new DatagramPacket(data2,
data2.length);
                try
                {
                    socket.receive(receivePacket);
                }
                catch (Exception e) {}

                    byte[] data3 = receivePacket.getData();
                    servo_val = (int) ((data3[0] & 0xff) << 8 | (data3[1] & 0xff));
                    motor_val = (int) ((data3[2] & 0xff) << 8 | (data3[3] & 0xff));
                    servo.setPulseWidth(servo_val);                    // pulse is between 1000
and 2000

                    motor.setPulseWidth(motor_val);
                }
                catch (Exception e) {Log.e("IOIO_thread","error: ", e);}
        }

        /** Not relevant to subclasses. */
        public synchronized final void abort()
        {
            abort_ = true;
            servo.close();
            motor.close();
        socket.close();
            if (ioio_ != null) {
                ioio_.disconnect();
            }
        }
}
```

```
package carl.IOIO_car;

import android.hardware.SensorManager;
import android.view.SurfaceView;

public class Main_thread extends Thread
{
	SurfaceView parent_context;
	SensorManager mSensorManager = null;
	Cam_thread the_cam;
	Sensors_thread the_sensors;
	IOIO_Thread ioio_thread_;
	String ip_address;
	Android_Activity the_app;

	public Main_thread(Android_Activity app, SurfaceView v, SensorManager m,
String ip)
	{
		super();
		parent_context = v;
		mSensorManager = m;
		ip_address = ip;
		the_app = app;
		the_cam = new Cam_thread(parent_context,ip_address);
		the_sensors = new Sensors_thread(mSensorManager,ip_address);
		ioio_thread_ = new IOIO_Thread(the_app, ip_address);
	}

    public void run()
    {
		ioio_thread_.start();
		the_cam.start_thread();
    }

    public void stop_simu()
    {
       the_cam.stop_thread();
       the_sensors.stop_thread();
		ioio_thread_.abort();
    }
}
```

```
package IOIO_car;

import java.net.DatagramPacket;
import java.net.DatagramSocket;
import java.net.InetAddress;

import android.hardware.SensorListener;
import android.hardware.SensorManager;
import android.util.Log;


@SuppressWarnings("deprecation")
public class Sensors_thread implements SensorListener
{
    final String tag = "Sensors";
    Android_Activity parent_context;
      InetAddress serverAddr;
      DatagramSocket socket;
      float x_0, y_0, z_0, x_A, y_A, z_A;
      short ix_0, iy_0, iz_0, ix_A, iy_A, iz_A;
      int size_p;

      SensorManager mSensorManager = null;
      String ip_address;

    public Sensors_thread(SensorManager sm, String ip)
      {
      super();
      mSensorManager = sm;
      ip_address = ip;

      try
      {
            serverAddr = InetAddress.getByName(ip_address);
            socket = new DatagramSocket();
      }
      catch (Exception exception)
      {
            Log.e(tag, "Error: ", exception);
      }
      }

     public void stop_thread()
     {
       socket.close();
     }

       private void send_data_UDP()
       {
             try
             {
             ix_0 = (short) (x_0);
             iy_0 = (short) (y_0);
             iz_0 = (short) (z_0);
```

```
            ix_A = (short) (x_A);
            iy_A = (short) (y_A);
            iz_A = (short) (z_A);

                  byte[] data = new byte[12];
                  data[0] = (byte) (ix_O >> 8);
                  data[1] = (byte) ix_O;
                  data[2] = (byte) (iy_O >> 8);
                  data[3] = (byte) iy_O;
                  data[4] = (byte) (iz_O >> 8);
                  data[5] = (byte) iz_O;
                  data[6] = (byte) (ix_A >> 8);
                  data[7] = (byte) ix_A;
                  data[8] = (byte) (iy_A >> 8);
                  data[9] = (byte) iy_A;
                  data[10] = (byte) (iz_A >> 8);
                  data[11] = (byte) iz_A;

                  try
                  {
                        size_p = data.length;
                        DatagramPacket packet = new DatagramPacket(data, size_p,
serverAddr, 9001);
                        socket.send(packet);
                  } catch (Exception e)
                  {
                        Log.e(tag, "Error: ", e);
                  }
            }
            catch (Exception exception)
            {
                  Log.e(tag, "Error: ", exception);
            }
      }

      public void onAccuracyChanged(int sensor, int accuracy) {}

      @Override
      public void onSensorChanged(int sensor, float[] values)
      {
            // TODO Auto-generated method stub
        synchronized (this)
        {
            if (sensor == SensorManager.SENSOR_ORIENTATION) {
                  x_O = values[0] *100;
                  y_O = values[1] *100;
                  z_O = values[2] *100;
            }
            if (sensor == SensorManager.SENSOR_ACCELEROMETER) {
                  x_A = values[0] *100;
                  y_A = values[1] *100;
                  z_A = values[2] *100;
            }
        }
       send_data_UDP();}}
```

Script PC

```
package pc.udp;

import java.io.ByteArrayInputStream;
import java.io.IOException;
import java.lang.Runnable;
import java.lang.Thread;
import java.net.DatagramPacket;
import java.net.DatagramSocket;
import java.net.InetAddress;

public class cam_thread_UDP implements Runnable
{
    int nb = 0;
    CAR_GUI car_state;
    Thread t;

      public static int HEADER_SIZE = 5;
      public static int DATAGRAM_MAX_SIZE = 1450;
      public static int DATA_MAX_SIZE = DATAGRAM_MAX_SIZE - HEADER_SIZE;

    public cam_thread_UDP(CAR_GUI gui)
    {
      car_state = gui;
       try
       {
            t = new Thread(this);
            t.start();
       }
       catch (Exception e){e.printStackTrace();}
    }

    public void run()
    {
        System.out.println("Menunggu Kamera Streaming ...");

        handleConnection_UDP();
    }

    public void handleConnection_UDP()
    {
            int current_frame = -1;
            int slicesStored = 0;
            byte[] imageData = null;
            DatagramSocket socket=null;

            try
            {
                  InetAddress serverAddr = InetAddress.getLocalHost();
                  System.out.println("PC IP address: " +
serverAddr.getHostAddress());
                  socket = new DatagramSocket(9000, serverAddr);
                  byte[] buffer = new byte[DATAGRAM_MAX_SIZE];
```

```
                DatagramPacket packet = new DatagramPacket(buffer,
buffer.length);

                while (true)
                {
                        socket.receive(packet);
                        byte[] data = packet.getData();
                        int frame_nb = (int)data[0];
                        int nb_packets = (int)data[1];
                        int packet_nb = (int)data[2];
                        int size_packet = (int) ((data[3] & 0xff) << 8 | (data[4]
& 0xff));

                        if((packet_nb==0) && (current_frame != frame_nb))
                        {
                                current_frame = frame_nb;
                                slicesStored = 0;
                                imageData = new byte[nb_packets * DATA_MAX_SIZE];
                        }

                        if(frame_nb == current_frame)
                        {
                                        System.arraycopy(data, HEADER_SIZE,
imageData, packet_nb * DATA_MAX_SIZE, size_packet);
                                        slicesStored++;
                        }

                        // If image is complete display it
                        if (slicesStored == nb_packets)
                        {
                                ByteArrayInputStream bis = new
ByteArrayInputStream(imageData);
                                car_state.set_image(bis);
                        }
                }
            } catch (IOException e)
            {
                    e.printStackTrace();
                    socket.close();
            }
    }
}
```

```
package pc.udp;

import java.io.ByteArrayInputStream;
import java.io.IOException;
import org.newdawn.slick.GameContainer;
import org.newdawn.slick.Graphics;
import org.newdawn.slick.Image;
import org.newdawn.slick.Input;
import org.newdawn.slick.SlickException;
import org.newdawn.slick.UnicodeFont;
import org.newdawn.slick.font.effects.ColorEffect;
import org.newdawn.slick.opengl.Texture;
import org.newdawn.slick.opengl.TextureLoader;
import org.newdawn.slick.state.BasicGameState;
import org.newdawn.slick.state.StateBasedGame;

public class CAR_GUI extends BasicGameState
{
	/** The ID given to this state */
	public static final int ID = 2;
	/** The font to write the message with */
	UnicodeFont uFont, uFont2;
	/** The image to be display */
	private Image logo, ima_up, ima_down, ima_left, ima_right;
	ByteArrayInputStream bis;
    Texture the_texture;

	GameContainer the_container;

	float x_O,y_O,z_O,x_A,y_A,z_A;
	float gas;
	int servo_val = 1500;
	boolean key_pressed=false, LEFT=false, RIGHT=false, UP=false, DOWN=false;

	cam_thread_UDP cam_thread;
	sensors_thread_UDP sensors_thread;
	ioio_thread_UDP IOIO_thread;


	/**
	 *      org.newdawn.slick.state.BasicGameState#getID()
	 */
	public int getID() {
		return ID;
	}

	/**
	 *
org.newdawn.slick.state.BasicGameState#init(org.newdawn.slick.GameContainer,
org.newdawn.slick.state.StateBasedGame)
	 */
	@SuppressWarnings("unchecked")
	public void init(GameContainer container, StateBasedGame game) throws
SlickException
	{
```

```
        String fontPath = "data-latin.ttf";
        uFont = new UnicodeFont(fontPath , 50, false, false);
        uFont.addAsciiGlyphs();
        uFont.addGlyphs(fontPath);
        uFont.getEffects().add(new ColorEffect(java.awt.Color.blue));

        uFont.loadGlyphs();

        uFont2 = new UnicodeFont(fontPath , 20, false, false);
        uFont2.addAsciiGlyphs();
        uFont2.addGlyphs(fontPath);
        uFont2.getEffects().add(new ColorEffect(java.awt.Color.red));

        uFont2.loadGlyphs();

        logo = new Image("CARL_Logo.jpg");
        ima_up = new Image("up_arrow.png");
        ima_down = new Image("down_arrow.png");
        ima_left = new Image("left_arrow.png");
        ima_right = new Image("right_arrow.png");

        ima_up.setAlpha((float) 0.5);
        ima_down.setAlpha((float) 0.5);
        ima_left.setAlpha((float) 0.5);
        ima_right.setAlpha((float) 0.5);

        cam_thread = new cam_thread_UDP(this);
        sensors_thread = new sensors_thread_UDP(this);
        IOIO_thread = new ioio_thread_UDP(this);
    }

    /**
     *
     * org.newdawn.slick.state.BasicGameState#render(org.newdawn.slick.GameContainer,
     * org.newdawn.slick.state.StateBasedGame, org.newdawn.slick.Graphics)
     */
    public void render(GameContainer container, StateBasedGame game, Graphics g)
    {
        try
        {
            if(bis != null)
            {
                the_texture = TextureLoader.getTexture("jpeg", bis);
                logo.setRotation(90);
                // to prevent it to overlap
                logo.setTexture(the_texture);
            }
        } catch (IOException e) {e.printStackTrace();}

        uFont.drawString(75,50, "KAMERA");
        g.drawImage(logo,175-(logo.getWidth()/2),260 -(logo.getHeight()/2));
        //uFont.drawString(72,450, "Sensors");
        uFont2.drawString(350,425, "Acceleration" + "\n x: " + x_A + "\n y: " +
y_A + "\n z: " +  z_A);
```

```
            uFont2.drawString(550,425, "Orientation" + "\n x: " + x_O + "\n y: " +
y_O + "\n z: " + z_O);

            uFont.drawString(440,50, "Motors");
            g.drawImage(ima_up,500,150);
            g.drawImage(ima_down,500,250);
            g.drawImage(ima_left,450,200);
            g.drawImage(ima_right,550, 200);

            uFont2.drawString(40,425, "Keterangan :");
            uFont2.drawString(40,450, " Tombol W = Maju" + "\n Tombol S = Mundur" +
"\n Tombol A = Kiri" + "\n Tombol D = Kanan");

            if(LEFT == true) ima_left.setAlpha((float) 1.0);
            else
            {
                  if(RIGHT == true)  ima_right.setAlpha((float) 1.0);
                  else
                  {
                        ima_left.setAlpha((float) 0.5);
                        ima_right.setAlpha((float) 0.5);
                  }
            }

            if(UP == true) ima_up.setAlpha((float) 1.0);
            else
            {
                  if(DOWN == true)  ima_down.setAlpha((float) 1.0);
                  else
                  {
                        ima_up.setAlpha((float) 0.5);
                        ima_down.setAlpha((float) 0.5);
                  }
            }
            uFont2.drawString(350,325, "PWM Servo \n" + IOIO_thread.servo);
            uFont2.drawString(550,325, "PWM Motor \n" + IOIO_thread.motor);
      }

      public void set_image(ByteArrayInputStream b)
      {
            bis =b;
      }

      public void set_sensors_values(float x_O2, float y_O2, float z_O2, float x_A2,
float y_A2, float z_A2)
      {
            x_O = x_O2;
            y_O = y_O2;
            z_O = z_O2;
            x_A = x_A2;
            y_A = y_A2;
            z_A = z_A2;
      }
```

```
    public void update(GameContainer container, StateBasedGame game, int delta)
    {
        synchronized(this)
        {
            RIGHT = false;
            LEFT = false;
            UP = false;
            DOWN = false;

            Input input = container.getInput();

            if(input.isKeyDown(Input.KEY_A) ^ input.isKeyDown(Input.KEY_D))
            {
                if (input.isKeyDown(Input.KEY_A))
                {
                    LEFT = true;
                }

                if (input.isKeyDown(Input.KEY_D))
                {
                    RIGHT = true;
                }
            }
            if(input.isKeyDown(Input.KEY_W) ^ input.isKeyDown(Input.KEY_S))
        {
                if (input.isKeyDown(Input.KEY_W) && DOWN==false)

                {
                    UP = true;
                }

                if (input.isKeyDown(Input.KEY_S) && UP==false)

                {
                    DOWN = true;
                }
            }
        }
    }

    /**
     * @see org.newdawn.slick.state.BasicGameState#keyReleased(int, char)
     */
    public void keyPressed(int key, char c)
    {
        if (key == Input.KEY_ESCAPE)System.exit(0);
    }
}
```

```
package pc.udp;

import java.net.DatagramPacket;
import java.net.DatagramSocket;
import java.net.InetAddress;

public class ioio_thread_UDP implements Runnable
{
    float gas;
    CAR_GUI car_state;
    Thread t;
    int size_p;
    short servo = 1500;
    short motor = 1500;


    public ioio_thread_UDP(CAR_GUI gui)
    {
        car_state = gui;
        try
        {
            t = new Thread(this);
            t.start();
        }
        catch (Exception e){e.printStackTrace();}
    }

    public void run()
    {
      System.out.println("IOIO thread waiting...");

            handleConnection_UDP();
    }

    public void handleConnection_UDP()
    {
      DatagramSocket socket=null;

      try
      {
            InetAddress serverAddr = InetAddress.getLocalHost();
            socket = new DatagramSocket(9002, serverAddr);
      }
      catch (Exception e){e.printStackTrace();}

      byte[] buffer = new byte[1];
      DatagramPacket packet_R = new DatagramPacket(buffer, buffer.length);
      InetAddress IP_phone=null;
      int port_phone = -1;

            try
            {
                  socket.receive(packet_R);
            }
```

```
        catch (Exception e) {}

        IP_phone = packet_R.getAddress();
        port_phone = packet_R.getPort();
        System.out.println("port android: " + port_phone);

while (true)
{
            if(car_state.LEFT == true)
            {
                if(servo >1000) servo = (short) (servo - 10);
            }
            else
            {
                if(car_state.RIGHT == true)
                {
                    if(servo <2000) servo = (short) (servo + 10);
                }
                else
                {
                    if(servo != 1500)
//default
                    {
                        if(servo<1500) servo = (short) (servo + 10);
                        else           servo = (short) (servo - 10);
                    }
                }
            }

            if(car_state.UP == true)
            {
                motor =  1950;
            }
            else
            {
                if(car_state.DOWN == true)
                {
                    motor =1400;
                }
                else
                {
                    motor = 0;
                }
            }

        if(port_phone > -1)
        {
            byte[] data2 = new byte[4];
            data2[0] = (byte) (servo >> 8);
            data2[1] = (byte) servo;
            data2[2] = (byte) (motor >> 8);
            data2[3] = (byte) motor;

            size_p = data2.length;
```

```
                    DatagramPacket packet_S = new DatagramPacket(data2, size_p,
IP_phone, port_phone);
                    try
                    {
                          socket.send(packet_S);
                    }
                    catch (Exception e){e.printStackTrace();}
              }

              try
              {
                    Thread.sleep(10);
              }
              catch (InterruptedException e) {e.printStackTrace();}
        }
    }
}
```

```
package pc.udp;

import org.newdawn.slick.AppGameContainer;
import org.newdawn.slick.GameContainer;
import org.newdawn.slick.SlickException;
import org.newdawn.slick.state.StateBasedGame;

public class Main_GUI extends StateBasedGame
{
	CAR_GUI car_state;

	public Main_GUI()
	{
		super("Iwan Fernando  Nim : 0812516");
		car_state = new CAR_GUI();
	}

	/**
	 *
org.newdawn.slick.state.StateBasedGame#initStatesList(org.newdawn.slick.GameContainer
)
	 */
	public void initStatesList(GameContainer container) {
		addState(car_state);
	}

	public static void main(String[] argv) {
		try {
			AppGameContainer container = new AppGameContainer(new
Main_GUI());
			container.setDisplayMode(700,546,false);
			container.start();
		} catch (SlickException e) {
			e.printStackTrace();
		}
	}
}
```

```
package pc.udp;
import java.io.IOException;
import java.lang.Runnable;
import java.lang.Thread;
import java.net.DatagramPacket;
import java.net.DatagramSocket;
import java.net.InetAddress;

public class sensors_thread_UDP implements Runnable
{
      float x_O,y_O,z_O,x_A,y_A,z_A;
    CAR_GUI car_state;
    Thread t;

    public sensors_thread_UDP(CAR_GUI gui)
    {
        try
        {
             t = new Thread(this);
             t.start();
        }
        catch (Exception e){e.printStackTrace();}
        car_state = gui;
    }

    public void run()
    {
      System.out.println("Monitor Sensor SAccelometer...");

      handleConnection_UDP();
    }

    public void handleConnection_UDP()
    {
            DatagramSocket socket=null;

            try
            {
                  InetAddress serverAddr = InetAddress.getLocalHost();
                  socket = new DatagramSocket(9001, serverAddr);
                  byte[] buffer = new byte[12];
                  DatagramPacket packet = new DatagramPacket(buffer,
buffer.length);

                  while (true)
                  {
                        socket.receive(packet);
                        byte[] data = packet.getData();

                        short nb = (short) ((data[0] & 0xff) << 8 | (data[1] &
0xff));
                        x_O = (float)nb / 100;

                        nb = (short) ((data[2] & 0xff) << 8 | (data[3] & 0xff));
```