TUGAS AKHIR – TE141599

Semi Autonomous Mobile Robot dengan Lengan untuk Mengambil Objek

Muhammad Qomaruzzaman
NRP 2211 100 145

Dosen Pembimbing
Ronny Mardiyanto, ST., MT., Ph.D
Dr. Ir. Djoko Purwanto, M.Eng.

JURUSAN TEKNIK ELEKTRO
FakultasTeknologiIndustri
Institut Teknologi Sepuluh Nopember
Surabaya 2016

FINAL PROJECT – TE141599

Semi Autonomous Mobile Robot with Arm to Pick Object

Muhammad Qomaruzzaman
NRP 2211 100 145

Advisor
Ronny Mardiyanto, ST., MT., Ph.D
Dr. Ir. Djoko Purwanto, M.Eng.

ELECTRICAL ENGINEERING DEPARTMENT
Faculty of Industrial Technology
Institut Teknologi Sepuluh Nopember
Surabaya 2016

SEMI AUTONOMOUS MOBILE ROBOT DENGAN LENGAN UNTUK MENGAMBIL OBJEK

TUGAS AKHIR

Diajukan Guna Memenuhi Sebagian Persyaratan
Untuk Memperoleh Gelar Sarjana Teknik
Pada
Bidang Studi Elektronika
Jurusan Teknik Elektro
Institut Teknologi Sepuluh Nopember

Menyetujui:

Dosen Pembimbing I,	Dosen Pembimbing II,
Ronny Mardiyanto, ST., MT.,Ph.D.	Dr. Ir. Djoko Purwanto, M.Eng.
NIP. 198101182003121003	NIP. 196512111990021002

SURABAYA
JANUARI 2016

SEMI AUTONOMOUS MOBILE ROBOT DENGAN LENGAN UNTUK MENGAMBIL OBJEK.

Muhammad Qomaruzzaman
2211100145

Dosen Pembimbing I : Ronny Mardiyanto, ST., MT., Ph.D
Dosen Pembimbing II :Dr. Ir. Djoko Purwanto, M.Eng.

ABSTRAK

Autonomous robot adalah robot dengan kemampuan untuk bergerak secara mandiri sesuai dengan instruksi yang diprogram. *Semi autonomous robot* adalah robot *autonomous*namun masih dapat dikendalikan secara manual.Kemampuan *semi autonomous* ditambahkan untuk mempercepat respon robot. Pada Tugas Akhir ini akan dirancang robot berlengan yang dikendalikan dari jarak jauh dengan kemampuan *semi autonomous* untuk mendekati objek. Robot dilengkapi dengan kamera dari dua sudut pandang. Satu kamera digunakan sebagai sensor pengindraan robot dan satu kamera untuk mempermudah navigasi robot. Pada tugas akhir ini pengguna akan memilih suatu objek dengan remot kontroler secara manual, kemudian robot akan secara otomatis mendekati objek. Robot berhenti 20cm didepan objek menggunakan sensor ultrasonik.Robot mengikuti objek menggunakan metode *Lucas-Kanade* dan kontrol proporsional untuk menghadapkan haluan robot ke objek. Dalam tugas akhir ini objek yang akan dikenali adalah objek dengan warna merah silindris dan kotak berwarna coklat menyerupai *C-4*.Berdasarkan pengamatan dan pengujian yang telah dilakukan, robot dapat mendekati objek secara otomatis dengan kecepatan 4,38cm/s. Jarak optimal robot dengan objek adalah 330cm. Beban yang dapat diangkat robot adalah 217gram.

Kata kunci : Robot mobil, penjajakan objek, pengolahan citra.

Halaman Ini Sengaja Dikosongkan

SEMI AUTONOMOUS MOBILE ROBOT WITH ARM TO PICK OBJECT.

Muhammad Qomaruzzaman
2211100145

Supervisor :Ronny Mardiyanto, ST., MT., Ph.D
Co-Supervisor :Dr. Ir. Djoko Purwanto, M.Eng.

ABSTRACT

Autonomous robot is robot with capability to do something on their own according to pre-programmed instructions.Semi autonomous robotis an autonomous robot but hence they can do something on their own, they can still be controlled by the operator. The most advantageous point using this metod is to improve user's convenience operating robot. The other advantageous point is user can take over robots control if anything go wrong. In this final project will be designed remotedly operated robot with arm andsemi autonomous capability. Robot designed with two cameras. One of them is used to aid the operator to navigate the robot through remote distance. And the other is used to lock or select certain object which is wanted to be approach. In this final project operator choose the object manually and the robot will approach it automatically. Robot tracks the object using Lucas-Kanade metode and proporsional to pan the robot.This final project define certain object to be tracked as premade red cilindricial C-4 like object. According to data that has been acquired during experimentation robot is able to approach object automatically at 4,38cm/s. The optimum distance between object and robot is 330cm and the robot's payload is 217gram.

Keywords :*Mobile robot, object tracking, image processing.*

Halaman Ini Sengaja Dikosongkan

KATA PENGANTAR

Segala puji bagi Allah raja seluruh penjuru alam semesta. Segala kasih sayang-Nya yang tak tercela telah memudahkan penulis dalam menyelesaikan tugas akhir berjudul **"*Semi Autonomus Mobile Robot* dengan Lengan untuk mengambil objek".**

Pengerjaan tugas akhir ini telah membuat penulis banyak belajar dan menerapkan kambali ilmu yang telah diperoleh selama masa perkuliahan. Penulis berharap dapat memberikan sumbangsih ilmu tersebut kepada generasi berikutnya maupun siapapun yang ingin mencuplik sedikit ilmu dari tugas akhir ini, sehingga terus terdapat kesinambungan pengembangan sistem yang serupa

Penulisinginmengabadikan ucapan terimakasih kepada seluruh pihak yang telah membantu penyempurnaan tugas akhir didalam kata pengantar yang singkat ini. Semoga kasih dan sayang dari Allah selalu tercurahkan kepada:

1. Nabi Muhammad SAW, meskipun tanpa kehadiran beliau secara fisik, beliaulah obat dan penghapus peluh penulis dalam menghadapi setiap tantangan pengerjaan tugas akhir ini.
2. Orang tua penulis yang tiada henti menemani penulis.
3. Bapak Ronny Mardiyanto, ST., MT.,Ph.D. dan Dr. Ir. Djoko Purwanto, M.Eng. yang tidak pernah lelah dan selalu sabar membimbing penulis. Beliau selalu bersedia meluangkan waktunya untuk memberikan dukungan baik moral maupun material dari awal hingga akhir penulisan tugas akhir.
4. Rekan ELVN sebagai teman seperjuangan dalam segala tantangan didunia pergulatan yang sekarang mulai gandrung disebut dengan masa mahasiswa.

Penulis menyadari bahwa tugas akhir ini penuh dengan kekurangan. Penulih sangat menghargai kritik dan saran untuk menutup kekurangan yang tercipta untuk disempurnakan.

Surabaya, Januari 2016

Halaman Ini Sengaja Dikosongkan

DAFTAR ISI

TABLE OF CONTENT

DAFTAR GAMBAR

DAFTAR TABEL

Halaman Ini Sengaja Dikosongkan

BAB I

PENDAHULUAN

1.1 Latar Belakang

Beberapa tahun yang lalu terjadi bencana terkait pembangkit listrik tenaga nuklir di Fukuhima, Jepang. Pemerintah Jepang menerjunkan robot penjelajah untuk menyisir daerah bencana tersebut. Robot ini diterjunkan untuk menggantikan manusia karena tingginya tingkat radiasi nuklir yang dapat membahayakan kesehatan manusia. Pada bulan Maret tahun 2011 terdapat bom berbentuk buku di Utan Kayu yang sempat meledak di salah satu tangan polisi. Hal ini di karenakan kecerobohan menghadapi benda yang berpotensi merugikan nyawa manusia. Penggunaan robot untuk menghadapi situasi berbahaya dapat terlihat pada robot Morolipi (Mobil Robot LIPI). Robot morolipi digunakan dalam upaya penjinakan bom. Dalam upaya tersebut robot Morolipi dioperasikan secara manual sepenuhnya.

Dalam kodisi tertentu manusia tidak dapat dilibatkan untuk melakukan hal-hal yang berbahaya sehingga diperlukan bantuan robot untuk menyelesaikan atau meminimalisasi risiko permasalahan yang dihadapi. Oleh karena itu penelitian ini dibuat untuk merancang robot *mobile* dan berlengan yang dapat dikendalikan dari jarak jauh untuk menjelajah dan mengambil benda. Pada robot *mobile* ini juga ditambahkan kamera untuk memilih benda sehingga robot dapat mendakati benda secara otomatis.

1.2 Perumusan Masalahan

Bedasarkan latas belakang di atas, dapat dirumuskan beberapa masalah, antara lain :

1. Bagaimana cara mengambil objek menggunakan lengan robot dari jarak jauh?
2. Bagaimana caramemilih objekmenggunakan kamera?
3. Bagaimana cara robot dapat mendekati objek tertentu secara semi autonomous?

1.3 Tujuan

Tujuan dari penelitian ini adalah:

1. Dapat mengambil objek menggunakan lengan robot dari jarak jauh.
2. Dapat memilih objek menggunakan kamera.

3. Dapat mendekatkan robot pada objek tertentu secara semi autonomous.

1.4 Batasan Masalah

Batasan masalah dari tugas akhir ini adalah sebagai berikut:

1. Robot tidak mengetahui letak halangan.
2. Robot berada pada bidang datar.
3. Objek yang didekati secara otomatis oleh robot adalah benda diam.
4. Tinggi objek tidak melebihi 10cm dan lebar benda tidak melebihi 5cm.
5. Pola atau warna benda tidak memiliki pola atau warna yang sama dengan daerah sekitar benda.

1.5 Metodologi Penelitian

Tugas akhir ini menggunakan metode penelitian sebagai berikut:

1. **Studi Literatur**
   Tahapan ini adalah segenap kegiatanpencarian dan pengkajian referensi yang berhubungan dengan topik tugas akhir. Referensi studi literatur didapatkan melalui buku, jurnal ilmiah, dan browsing melalui internet yang berhubungan dengan judul tugas akhir ini. Studi literatur dilakukan secara beriringan dengan penelitian.
2. **Perancangan Robot**
   Pada tahap ini akan dirancang robot yang dapat menjelajah dan mengambil objek dari jarak jauh. Perancangan ini dilakukan dengan studi literatur sebelumnya. Perancangan ini akan meliputi perancangan *hardware* dan *software* robot. Perancangan *hardware* meliputi mekanisme robot seperti lengan robot beserta motor servo dan juga motor penggerak robot. Perancangan *software* robot akan dilakukan pada Arduino. Segenap kontrol seperti inverse kinematic ataupun pengontrol kecepatan robot akan dilakukan pada tahapan ini melalui perancangan *software.*
3. **Pengujian dan Analisa Data**
   Pada tahapan ini akan dilakukan akan dilakukan berbagai pengujian terhadap robot untuk melihat respon robot. Jarak kontrol dan kecepatan respon robot adalah salah satu data yang

akan diuji. Pada pengujian jarak kontrol akan dilakukan pengujian untuk mengetahui respon robot terhadap jarak robot dari kontroler. Pengujian kecepatan robot adalah pengujian untuk mengetahui kecepatan respon robot terhadap perintah yang diberikan.

4. **Penarikan Kesimpulan dan Saran**
   Kesimpulan ditarik dari pengujian, analisa data, dan juga referensi. Pada kesimpulan akan ditarik garis besar jawaban dari permasalahan yang dianalisis. Selain itu, juga akan diberikan saran sebagai masukan berkaitan dengan apa yang telah dilakukan.Kesimpulan dan saran dapat digunakan untuk pengembangan penelitian selanjutnya.
5. **Penyusunan Buku Tugas Akhir**
   Tahap penyusunan buku tugas akhir akan dilakukan beriringan dengan penelitian. Penyusunan buku tugas akhir adalah bentuk laporan tertulis dari penelitian. Penulisan buku tugas akhir dilakukan dengan sistematika penulisan yang telah ditentukan.

1.6 Sistematika Penulisan

Laporan tugas akhir ini ditulis berdasarkan format yang telah ditentukan oleh Jurusan Teknik Elektro Institut Teknologi Sepuluh Nopember. Tugas akhir ini terdiri dari lima bab yang akan dijelaskan pada poin di bawah.

- Bab 1 : Pendahuluan
  Bab ini menjelaskan latar belakang, perumusan masalah, tujuan, sistematika penulisan, metodologi, dan relevansi tugas akhir.
- Bab 2 : Dasar Teori
  Bab ini menjelaskan tentang berbagai macam teori pendukung dalam penulisan tugas akhir. Bab ini meliputi dasar teori dari Inverse kinematics, radio kontroler, ESC, PWM, motor servo, GPS, rotary encoder, sensor ultrasonik, dan multiplexer
- Bab 3 : Perancangan Sistem
- Bab 4 : Pengujian Analisis
- Bab 5 : Penutup

1.7 Relevansi

Mata kuliah pendukung tugas akhir ini adalah Sistem Mikroprosesor dan Mikrokontroler, Robot Industri, dan Pengindraan

Visual. Tugas akhir ini memiliki relevansi dengan *tracking system* menggunakankamera.

Hasil dari tugas akhir ini diharapkan dapat memberikan sumbangsih penelitian dalam bidang elektronika dan juga bidang penelitian menggunakan kamera untuk aplikasi *tracking***.**

BAB II

DASAR TEORI DAN TINJAUAN PUSTAKA

Pada bab ini akan dijelaskan berbagai macam dasar teori dan tinjauan pustaka yang digunakan dalam perancangan sistem. Dalam dasar teori akan dijelaskan berbagai macam teori penunjang dalam penulisan dan perancangan tugas akhir. Sedangkan dalam tinjauan pustaka akan dijelaskan tentang sistem yang berhubungan tugas akhir ini oleh penulis-penulis sebelumnya.

2.1 Dasar Tori

2.1.1 *Robot Kinematics*

Kinematika adalah ilmu yang mempelajari gerak tanpa mengindahkan gaya yang menyebabkan gerakan tersebut [1]. Robot kinematics (kinematika robot) adalah suatu disiplin ilmu yang mengaplikasikan geometri untuk mempelajari gerakan suatu titik yang menyusun struktur sistem robot.Geometri yang dimaksud adalah setiap bagian robot dimodelkan sebagai *rigid bodies*(benda tegar)dan setiap sendi bagian robot menghasilkan gerakan translasi atau rotasi secara murni. *Rigid bodies*(benda tegar)adalah suatu kesatuan benda secara ideal yang bentuk dan ukurannya tidak berubah ketika mendapatkan suatu gaya tertentu.Setiap robot bagian robot secara teori dianggap sebagai benda tegar, meskipun pada kenyatannya tidaklah sama. Robot kinematics mempelajari hubungan antara dimensi dan setiap sendi (*joint*) dari beberapa bagian (*link*) kinematika dan posisi, kecepatan dan akselerasi dari setiap bagian robot untuk merencanakan kontrol pergerakan dan menghitung gaya dan torsi aktuator.

Robot terdiri dari link yang dihubungkan oleh joint. Biasanya setiap joint dilengkapi dengan instrument untuk mengukur posisinya relative terhadap posisi link yang diukur. Pada joint yang berputar (*rotary* atau *revolute*) perpindahan poisi link disebutdengan sudut joint (*joint angles*). Beberapa robot juga terdiri dari sendi yang bergerak secara translasi tau prismatic. Perpindahan gerakan pada joint yang seperti ini biasanya disebut *joint offset.* Setiap bagian joint akan menghasilkan tingkat kebebasan robot dalam bergerak. Dengan bertambahnya maka bertambah pula gerakan robot. Jumlah yang menunjukkan tingkat keluesan robot biasanya di sebut sebagai *Degree of Freedom (DOF*)[2].

Alat yang sangat fundamental pada kinematika robot adalah persamaan kinematika dari rantai kinematika yang menyusun robot. Persamaan kinematika juga digunakan pada biomekanik dan juga animasi komputer. Ada dua persamaan utama yang biasa digunakan pada kinematika robot, yaitu *Forward kinematics* (Kinematika Maju) dan *Reverse kinematics* (Kinematika Mundur). Forward kinematics menggunakan persamaan kinematika untuk menghitung posisi ujung (*end-effector*) dari nilai yang telah ditentukan pada setiap parameter sendi. Proses sebaliknya yaitu menghitung posisi *joint*untuk menentukan posisi *end-effector*. Proses tersebut biasa dikenal dengan Reverse kinematics. Dimensi robot dan persamaan kinematika menentukan volum jangkauan robot yang biasa disebut workspace [2].Pada robot kinematics sering ditemui istilah *DH Parameter (Denavit-Hartenberg Parameter).* DH Parameter adalah suatu parameter yang terkait dengan kovensi referensi setiap link robot.

2.1.2*Radio Controller*

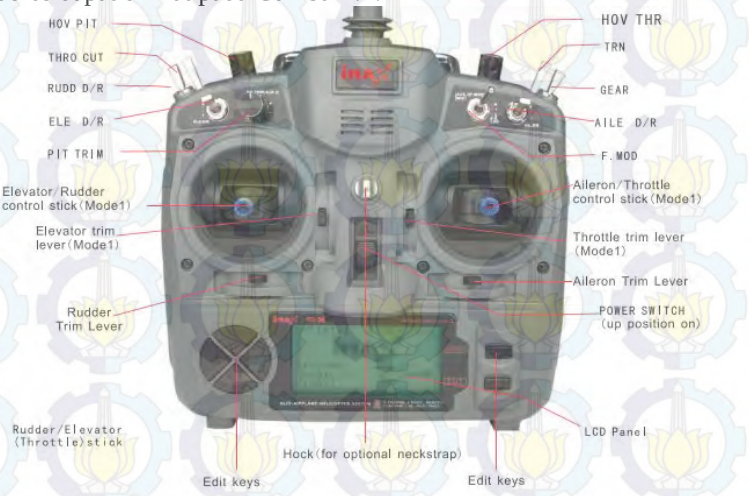
Radio controller (pengontrol radio) banyak digunakan di kehidupan sehari hari seperti, pembuka pintu garasi, remot kontrol televisi, dan juga mobil mainan. Prinsip dasar kerja *radio controller* adalah sama, yaitu transmitter mengirim sinyal biner termodulasi, kemudian *radio receiver*(penerima sinyal radio) membaca sinyal apakah sinyal tersebut sesuai dengan frekuensi yang diinginkan, kemudian penerima sinyal menerjemahkannya untuk lebih mudah diolah [3].

Radio controller adalah suatu alat yang menggunakan sinyal radio untuk mengontrol suatu peranti dari jarak jauh. Pengontrol radio biasanya digunakan untuk mengendalikan berbagai macam peranti dari sebuah pengontrol yang dapat dibawa-bawa (*mobile*). Berbagai macam kepentingan penelitian ilmiah, industri, dan militer juga memanfaatkan pengontrol radio. Penelitian ini digunakan pengontrol radiountuk mengendalikan robot.Pengontrol radioadalah *radio transmitter* (pengirim sinyal radio) yang biasanya terdiri dari berbagai macam tuas, saklar, dan juga tombol. Penerima sinyal radio dari pengontrol radio dipasang pada robot. Penerima sinyal radio menerima sinyal dari Pengirim sinyal radio untuk kemudian dilanjutkan menuju motor servo, motor DC, atau berbagai keperluan lainnya.

Pengirim sinyal radio mengirimkan sinyal menuju penerima sinyaldengan modulator dan multiplexer. Sinyal yang dikirim oleh

pengirim sinyal radio kemudian diterjemahkan oleh penerima sinyalradio dengan mendemodulasi sinyal dan membagi sinyal kedalam beberapa kanal. Sinyal tersebut juga diterjemahkan kedalam sinyal yang biasa digunakan pada motor servo pada umumnya. Pengontrol radio memiliki frekuensi kerja spesifik namun terkadang dapat terjadi interferensi. Pada beberapa bidang seperti industri dengan alat berat, interferensi radio bisa sangat berbahaya dan merugikan. Maka dari itu pada pengontrol radio pada bidang industri sinyal yang dikirim oleh pengirim sinyal biasanya disertai kode tertentu untuk diterjemahkan penerima sinyal. Akan tetapi, pengontrol radio pada kelas hobi atau mainan biasanya memiliki pilihan frekuensi yang sangat luas, sehingga jarang terjadi interferensi [3].

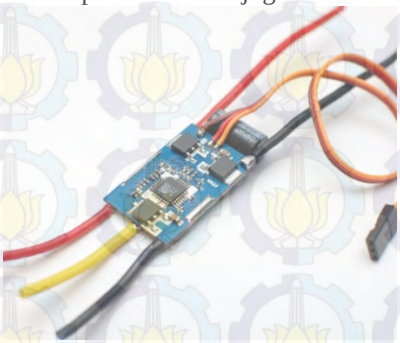
Pada penelitian ini terdapat banyak komunikasi sinyal radio yang akan digunakan. Diantara alat komunikasi radio yang akan digunakan dalam penelitian ini adalah pengontrol radio yang sering digunakan pada mobil mainan. Pengontrol radio ini dipilih karena fungsionalitasnya yang tinggi serta ergonomis. Pengontrol radio ini didesain untuk dapat dikembangkan dengan mudah oleh para *hobbies.* Komunikasi yang lain yang menggunakan radio adalah kamera nirkabel. Gambar pengontrol radio yang sering digunakan para *hobbies* dapat dilihat pada Gambar2.1.



Gambar 2.1 *Radio controller*

2.1.3ESC *(Electronic Speed Control)*

Laju robot dikontrol menggunakan *ESC (Electronic Speed Control)* atau pengontrol kecepatan elektronik. Tenaga yang diberikan pada ESC sebanding dengan tuas pada pengontrol radio. Semakin jauh tuas di dorong, maka semakin cepat pula robot melaju. Tegangan pengontrol kecepatan adalah sinyal pulsa dengan berbagai macam output dan menggunakan transistor yang telah di seleksi untuk menghasilkan transisi sinyal yang halus dan efisiensi tinggi. ESC juga dapat digunakan sebagai rem magnetic pada motor, sehingga pengontrol kecepatan secara elektronis lebih unggul dari pada pengontrol kecepatan secara mekanis. Pengontrol kecepatan secara mekanis menggunakan jaringan resistor. Jaringan resistor ini disambungkan secara bergantian dengan cara memutar controller sehingga elektroda kontroler tersambung dengan salah satu resistor. Reaksi dari pengontrol kecepatan mekanis lebih lambat dari pada pengontrol kecepatan elektronis karena didalam pengontrol kecepatan mekanis terdapat motor servo untuk memutar jaringan elektroda dan menghubungkannya dengan jaringan resistor. Pengontrol kecepatan mekanis juga lebih cepat panas akibat dari daya yang terbuang pada resistor. Pengontrol kecepatan mekanis juga lebih mudah kotor.



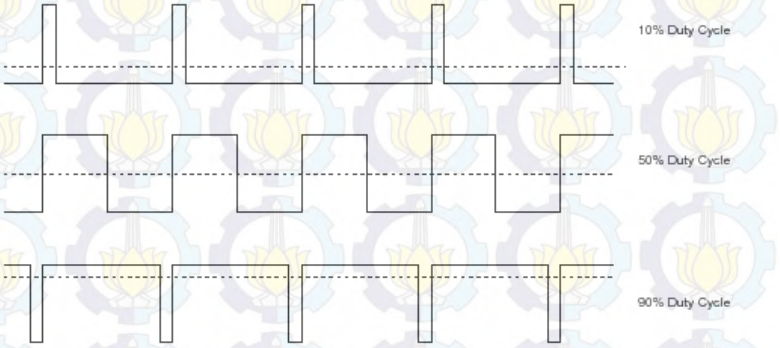
Gambar 2.2 ESC

Pada penelitian ini ESC digunakan untuk mengatur arah grak robot. Selain untuk mengatur laju dan arah gerak robot, ESC dapat digunakan sebagai pengatur laju kecepatan robot dengan mudah. Keuntungan lain dari ESC adalah pilihan spesifikasinya yang sangat bermacam-macam.

2.1.4 Pulse Width Modulation

PWM adalah suatu sinyal yang dikirim dengan frekuensi tetap namun dapatmemiliki panjang pulse yang berbeda-beda dalam setiap periodenya. Perbedaan ini biasanya disebut dengan *Duty cycle*, yaitu perbandingan lama pulse dengan keseluruhan periode sinyal. *Duty cycle* dinyatakan dalam persen. PWM adalah istilah yang biasanya disebut sebagai sinyal keluaran (*output*) analog [4]. Caranya adalah dengan mengubah tegangan rata-rata setiap periodenya. PWM dengan kondisi tertentu dapat dikonversi menjadi Pulse Position modulation mengingat karakteristiknya yang sama.

Pulse Position modulation lebih sering dikenal dengan PPM. PPM banyak digunakan dalam sinyal kontrol motor servo. PPM biasanya di kaitkan dengan Pulse Width Modullation atau PWM. PPM memiliki sinyal high yang sama, namun karena dimungkinkan jarak antar sinyal high berbeda-beda, maka PPM tidak memiliki frekuensi tetap. Jarak antar pulse (interval) yang menentukan data yang dibawa[5].



Gambar 2.3 Sinyal PWM

Penelitian ini menggunakan PWM sebagai sinyal kontrol ESC dan juga motor servo. PWM sangat banyak ditemukan sebagai pengontrol kecepatan motor DC. PWM mengatur kecepatan motor DC menggunakan tegangan rata-rata output dan frekuensi tertentu yang telah disesuaikan. Akan tetapi, dengan menggunakan PWM yang ditambahkanESC maka penggunakan frekuensi PWM untuk motor DC dan motor servo dapat disamakan.

2.1.5 *Servo Motor*

Servo Motor(motor servo) adalah suatu aktuator yang dapat dikontrol dengan suatu sudut perputaran atau perpindahan linear. Motor servo mempunyai konfigurasi gear sehingga motor servo dapat mudah dikontrol. Motor servo pada umumnya dapat dikontrol mulai dari sudut 0 hingga 180 derajad. Ada pula motor servo yang dapat dikontrol secara kontinyu dengan kecepatan yang dapat diatur pula[5].

Servo motor dikontrol menggunakan sinyal PWM dengan frekuensi 50Hz. Panjang sinyal pulse yang mengontrol bermacam-macam. Standard *duty cycle* untuk mengontrol servo adalah 5% sampai dengan 10%, namun biasanya sinyal pulse yang mengontrol motor servo berkisar 3.5% sampai dengan 11.5% [5].

Pada penelitian ini digunakan beberapa motor servo sebagai penggerak lengan robot. Dengan motor servo, setiap sudut bagian lengan robot dapat dihitung dengan mudah. Kelemahan dari motor servo adalah harganya yang mahal seiring dengan bertambanhnya torsi yang dapat di topang motor servo. Keterbatasan torsi dan ketidak terjangkauan harga motor servo seringkali mengakibatkan berubahnya konstruksi lengan robot.

2.1.6 GPS (*Global Positioning System*)

Gambar 2.4 3DR GPS Ublox

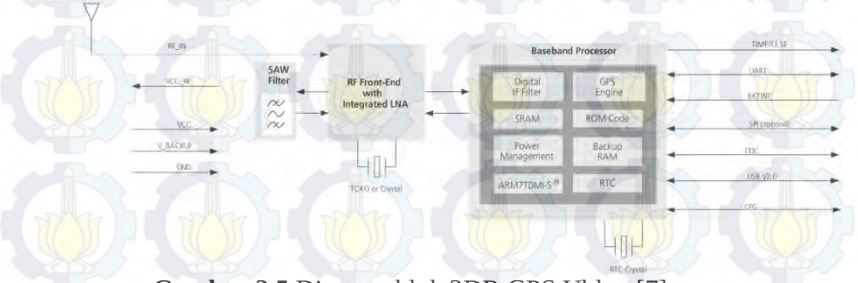
Global positioning system atau disingkat GPS adalah suatu teknologi navigasi yang menyediakan informasi posisi dan waktu kapan saja, diamana saja atau dekat dengan bumi, dalam cuaca apapun selama tidak ada yang menghalagi pandangan empat atau lebih satelit GPS. GPS adalah teknologi yang dibuat dan dirawat oleh pemerintah Amerika Serikat serta menjadikannya sebagai sarana yang bebas di akses menggunakan penerima sinyal GPS. GPS sangat banyak digunakan dalam bidang militer, komersial, dan juga seluruh pengguna

komersial diseluruh dunia. GPS adalah suatu teknologi yang akan menunjukkan posisi penerima sinyal GPS yang ada pada suatu benda.

GPS dapat dipasang pada alat apapun atau benda apapun. GPS dapat dipasang pada benda bergerak atau benda diam, semisalnya pada mobil atau pada telepon genggam atau pun alat khusus pelacak lokasi. GPS juga dapat mendeteksi lokasi benda bergerak seperti mobil, sehingga dengan GPS akan dapat diketahui posisi, arah, dan kecepatan benda secara langsung ataupun tertunda, bahkan posisi dapat diketahui secara tiga dimensi. Data pada GPS dapat digunakan untuk mengetahui pergerakan atau rute suatu benda [6].

Untuk mengetahui lokasi suatu benda, penerima sinyal GPS memerlukan dua data, yaitu data waktu dan posisi satelit GPS. Satelit GPS mempunyai jam atom, yaitu jam dengan akurasi yang sangat tinggi. GPS memancarkan sinyal data waktu data tersebut dikirim. Selisih waktu sinyal terkirim dan sampai dapat digunakan untuk menghitung jarak penerima sinyal GPS dengan satelit pengirim. Satelit GPS juga mengetahui posisi orbitnya. Dengan memperoleh data kapan sinyal dikirim satelit GPS dan diterima oleh penerima sinyal GPS, *receiver* GPS dapat mengetahui jarak dan posisi*receiver*dari satelit. Data dari 4 satelit dapat digunakan untuk mengetahui lokasi receiver secara 3D, yaitu lokasi *longitude* (lintang), *latitude* (bujur), dan *altitude* (ketinggian),

Penelitian ini menggunakan data GPS yang diolah Arduino untuk menentukan posisi Robot. GPS juga digunakan untuk mengetahui rute yang telah ditempuh robot dan juga merencanakan rute. GPS sangat berguna untuk melacak jejak dan posisi robot. GPS yang digunakan adalah 3DR GPS Ublox. Berikut ini adalah blok diagram 3DR GPS Ublox

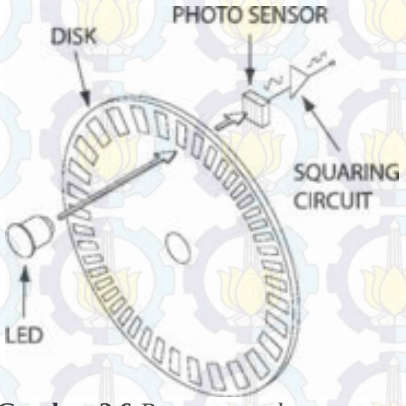


Gambar 2.5 Diagram blok 3DR GPS Ublox [7]

2.1.7 *Rotary Encoder*

Rotary encoder adalah suatu peranti yang dapat memonitor gerakan dan posisi.Rotari Encoder biasanya menggunakan sensor optik untuk menghasilkan sinyal pulsa. Pada tugas akhir ini *rotary encoder* digunakan untuk mengetahui kecepatan.

Pada *rotaryencoder* terdapat LED dan fototransistor. Kemudian diantara fototransistor dan LED terdapat piringan dengan lubang-lubang. Piringan berputar bersamaan dengan poros, sedangkan fotodioda dan LED diam sejajar mengapit piringan. Pada saat piringan berbutar fotodioda akan menangkap sinyal pulsa dari LED. Sinyal pulsa ini terbentuk karena LED tertutup dan terbuka oleh lubang piringan, sehingga fotodioda menerima cahaya LED dengan hidup dan mati secara bergantian.



Gambar 2.6 *Rotary encoder*

Untuk mengetahui kecepatan robot digunakan perhitungan frekuensi sebagai berikut

$$V = \frac{S_1 - S_0}{|T_1 - T_0|} \quad (1)$$

Keterangan :
V= Kecepatan
S_1 = Jarak tempuh sekarang.
S_0= Jarak tempuh sebelumnya.
T_1= Waktu sekarang.
T_0= Waktu sebelumnya.

2.1.8 Sensor Ultrasonik

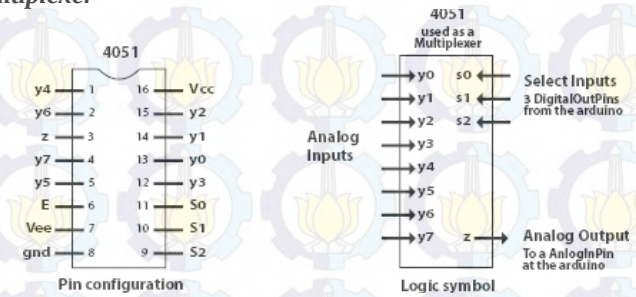
Sensor ultrasonik didunia pasaran elektronik juga disebut dengan *Ping sensor* atau *Paralax sensor*. Sensor ultrasonik adalah sensor yang memancarkan sinyal ultrasonik kemunian menangkap kembali sinyal tersebut. Sensor ultrasonik dapat digunakanuntuk menghitng jarak dengan mengukur jeda waktu antara sinyal yang dikirim dan diterima.

Pada tugas akhir ini sensor ultrasonik digunakan untuk menghitung jarak robot dengan halangan didepan robot. Jarak dapat dihitung menggunakan perhitungan sebagai berikut.

$$S = \frac{V * t}{2} \quad (2)$$

Keterangan :
S = Jarak antara sensor dengan halangan dihitung dalam satuan centimeter.
V=29 cm/s, yaitu kecepatan suara diudara.
t= Waktu yang ditempuh mulai dari sinyal dikirim sampai dengan sinyal kembali dihitung dalam satuan microsecond

2.1.9 *Multiplexer*



Gambar 2.7 Multiplexer

Pada tugas akhir ini digunakan *multiplexer* analog yaitu IC 4051. IC ini digunakan untuk memilih salah satu sinyal diantara dua sinyal videoyang akandikirim ke pengirim sinyal video. IC ini memiliki tiga selector untuk delapan input dan satu ouput. Tabel kebenaran untuk IC ini dapat dilihat pada gambar dibawah ini.

TRUTH TABLE
HC/HCT4051

INPUT STATES				"ON" CHANNELS
ENABLE	S_2	S_1	S_0	
L	L	L	L	A0
L	L	L	H	A1
L	L	H	L	A2
L	L	H	H	A3
L	H	L	L	A4
L	H	L	H	A5
L	H	H	L	A6
L	H	H	H	A7
H	X	X	X	None

X = Don't care

Gambar 2.8 Tabel kebenaran IC 4051

2.2 Tinjauan Pustaka

2.2.1 Pengindraan komputer

Pengindraan computer (*Computer Vision*) adalah serangkaian tranformasidata dari gambar menjadi sebuah representasi baru atau. Serangkaian tranformasi tersebut memiliki tujuan untuk dicapai. Informasi yang diambil pada pengindraan computer sangat bermacam-macam. Informasi tersebut dapat berupa kamera pada jalan raya atau sebuah foto keluarga.

Pengindraan manusia yang telah diproses oleh otak memiliki berbagai macam informasi yang sangat kompleks. Manusia dapat memperkirakan bola yang melambung pada lapangan sepakbola akan jatuh pada titik tertentu, bahkan memperkirakan titik tersebut dengan sangat akurat. Manusia dapat mengenali masing-masing wajah dari seribu orang rekan baik dalam satu kelompok mamupun dalam lingkungan yang sangat berbeda. Tanpa menyadari kerumitan dari informasi yang tersebut, terkadang membuat manusia terlena tentang bagaimana otak manusia dapat bekerja mengolah data-data tersebut.

Berbeda dengan manusia, computer hanya melihat sekumpulan warna pixel demi pixel. Sekumpulan warna tersebut tidak lain hanyalah kombinasi dan kumpulan angka-angka. Manusia member nama setiap warna dengan putih, hitam, abu-abu, dan sebaginya. Namun, computer melihat warna-warna tersebut dengan kumpulan angka yang tidak pernah sama meskipun manusia menggap warna itu adalah sama. Pengindraan computer adalah ilmu yang mempelajari bagai mana caar menerjemahkan setiap data dari suatu gambar menjadi representasi yang lain untuk memenuhi suatu tujuan. Pengindraan computer menjadi ilmu yang sangat luas hanya dengan membahas pertanyaan "bagaimana".

Dalam tugas akhir ini setiap proses yang melibatkan pengindraan computer atau pengolahan citra dilakukan menggunakan *library* dari OpenCV. OpenCV adalah suatu *library open source* yang dapat digunakan siapapun tanpa dikenakan biaya. OpenCV ditulis dalam bahasa pemrograman C/C++. OpenCV dikembangkan untuk menambah efisiensi dalam perhitungan pengindraan computer. OpenCV juga memiliki fokus pengembangan terhadap pengolahan gambar secara langsung dengan sumber dari perangkat seperti kamera. Dengan Fasilitas yang ditawarkan OpenCV, OpenCV adalah suatu alat penunjang yang sangat pas dan sangat penting dalam penyelesaian tugas akhir ini.

2.2.2 Metode penjajakan Lukas-Kanade

Tracking(penjajakan) adalah suatu cara untuk mempertahakan perbedaan sesuatu konstanta diatara dua atau lebih komponen. Ketika suatu pengolahan citra menerima data berupa video atau serangkaian gambar berurutan, terkadang terdapat suatu bagian tertentu pada video tersebut. Aplikasi penjajakan pada pengolahan citra sudah sangat populer. Contohnya, terdapat kamera yang ditempatkan pada suatu helicopter mainan yang diterbangkan diatas lapangan sepakbola. Terdapat suatu bagian pada gambar yang diterima helicopter tersebut, misalkan bola, yang ingin terus diikuti menggunakan kamera, sehingga helicopter dapat bergerak secara otomatis mengikuti bola.

Penjajakan biasanya melibatkan proses kamera untuk mengenali suatu suatu benda tertentu. Pengenalan benda dapat melibatkan berbagai macam proses, seperti histogram. Untuk dapat melakukan penjajakan tanpa proses kamera mengenali, objek tertentu maka dibutuhkan metode untuk mencari titik-titik kunci dari sebuah

gambar. Titik-titik kunci itu adalah suatu ciri has gambar yang akan mudah ditemukan pada setiap urutan gambar dalam suatu video. OpenCV telah menyediakan suatu metode untuk mencapai tujuan tersebut, yaitu Lucas-Kanade dan Horn-Schunk. Terkadang metode tersebut dikenal dengan metode *sparse* atau *dense*. [8] Pada menyelesaian tugas akhir ini digunakan metode Lucas-Kanade.

Pada penentuan titik-titik kunci suatu gambar, titik kunci tersebut haruslah suatu titik yang unik, atau mendekati uink. Titik-titik kunci ini akan mudah ditemukan pada setiap frame pada suatu video. Titik-titik kunci tidak mungkin terdapat pada suatu gambar dengan warna uniform, seperti dinding polos berwarna putih. Titik kunci juga tidak mungkin terdapat pada suatu gambar dengan pola tertentu yang sama. Titik-titik ini terkadang lebih dikenal dengan istilah *features* (fitur)atau *corners* (pojok, bukan tepi). Fungsi untuk menentukan *corners* ini sudah disediakan oleh OpenCV.

Metode Lucas-Kanade memanfaatkan intensitas *brightness* dari gambar*features*. Metode Lucas-Kanade dapat dilakukan pada gambar berwarna ataupun pada gambar abu-abu. Gambar abu-abu memiliki satu kanal warna, sedangkan gambar warna memiliki tiga kanal warna (kanal R, kanal G, kanal B). Oleh karena itu metode Lucas-Kanade dapat bekerja lebih cepat pada gambar abu-abu karena hanya memerlukan satu iterasi. Gambar abu-abu dalam satu *pixel* dapat didapatkan dengan persamaan seperti dibawah:

$$Gray = \frac{R + G + B}{3} \quad (3)$$

Keterangan:
Gray = Gambar abu-abu.
R= *G*ambar kanal merah (*Red*)
G= *G*ambar kanal hijau (*Green*)
B= *G*ambar kanal biru (*Blue*)

Metode Lucas-Kanade tidaklah mengenali objek secara utuh untuk diambil titik fiturnya, melaikan hanya sebagian kecil dari objek dengan. Bagian tersebut ditentukan oleh ukuran jendela yang disebut *patch.* Fitur lebih mudah ditemukan pada ukuran *patch* yang kecil.

Pada gambar bergerak, seperti video, terdapat urutan gambar sedemikian hingga seolah-olah gambar tersebut bergerak. Lucas-Kanade melakukan penjajakan pada gambar bergerak dengan cara mencari titik fitur yang sama dalam suatu *patch*urutangambar. Sehingga apabila suatu fitur tidak diemukan pada *patch* maka

penjajakan tidak dapat dilanjutkan. Fitur dapat hilang dikarenakan bergerak terlalu cepat sehingga keluar dari *patch* atau intensitas *brightness* fitur berubah. *Patch* yang berukuran besar dapat mendetaksi pergerakan fitur yang besar Jika fitur ditemukan maka *patch*akan diperbarui mengikuti fitur. Apabila piksel di lokasi (x,y,t) dengan intensitas I (x,y,t) melakukan perpindahansebanyak Δx, Δy, dan Δt pada frame berikutnya, ketetapan brightnessdapat diberikan [9]

$$I(x, y, t) = I(x + \Delta x, y + \Delta y, t + \Delta t) \quad (4)$$

Kelemahan dari memanfaatkan fitur dari gambar untuk aplikasi *tracking* adalah apabila ada objek yang besar bergerak melintasi kamera. Objek tersebut akan memindah fitur terhadap fitur yang terdekat dengan objek tersebut. Lukas-Kanade dalam aplikasinya memiliki asumsi sebagai berikut:

1. *Brightness Constancy.* Artinya penampilan objek tidak berubah, meskipun bergerak dari frame ke frame berikutnya. Pada gambar *grayscale* yang tidak berubah adalah kecerahan pixel suatu objek.
2. *Temporal persistence.* Artinya objek tidak bergerak dengan cepat.
3. *Spatial coherence.* Artinya apabila ada dua fitur yang digunakan dalam satu objek, maka kedua fitur tersebut haruslah pada permukaan yang sama pada objek sebenarnya.

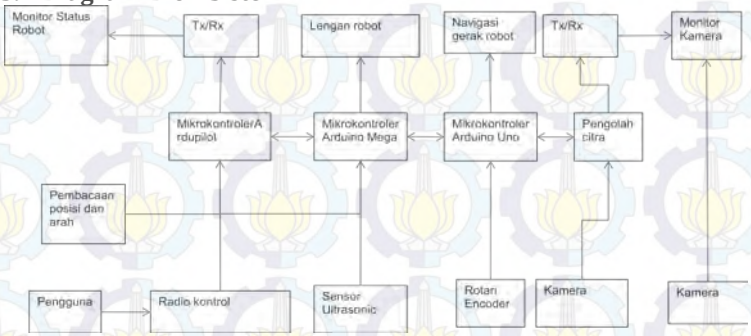
Halaman Ini Sengaja Dikosongkan

BAB III

PERANCANGAN SISTEM

Sistem yang dirancang dalam tugas akhir ini terdiri dari banyak bagian, seperti pengontrol radio, robot dengan sistem roda mirip tank, kamera, lengan robot, dan lain lain. Pengontrol radio digunakan sebagai antarmuka pengguna dengan robot untuk mengendalikan robot secara manual. Setiap bagian dari perancangan sistem akan dijelaskan di dalam bab ini secara lebih terperinci.

3.1 Diagram Blok Sistem



Gambar 3.1 Diagram Sistem Robot

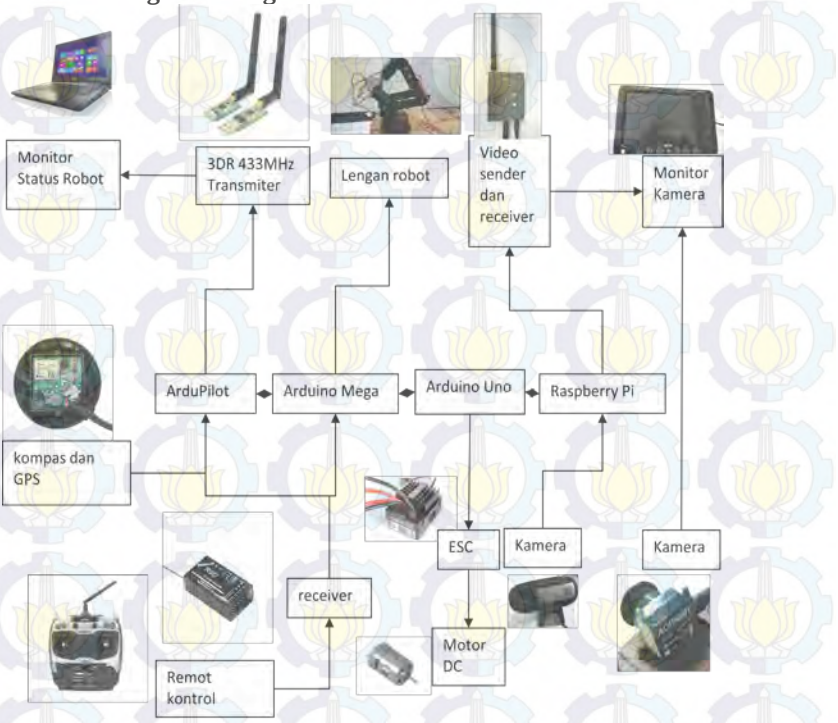
Pada sistem ini terdapat beberapa mode pengendalian. Robot dikendalikan oleh user menggunakan pengontrol radio. Dengan menggunakan pengontrol radio, pengguna dapat memilih mode pengendalian. Mode pengendalian tersebut adalah, mode manual, mode memilih objek, dan mode mendekati objek secara otomatis. Terdapat pula mode tambahan yaitu mode mengikuti koordinat peta dan pulang secara otomatis menggunakan GPS.Setiap data pada pengontrol radio dikirim, kemudian diterima oleh penerima sinyal radio. Data yang diterima penerima sinyal radio dikirimkan kepada unit proses.

Pada sistem ini terdapat tiga unit proses. Dua unit proses digunakan untuk mengelola data dari penerima sinyal radio, yaitu Arduino Uno dan Arduino Mega. Arduino Mega mengolah data yang diterima dari penerima sinyal radio menjadi gerakan lengan robot dengan *inverse kinematics*. Arduino Uno mengolah data yang diterima

dari receiver menjadi gerakan navigasi dengan cara*differential steer*. Unit proses yang terakhir adalah Raspberry Pi. Raspberry Pi digunakan sebagai pengolah citra untuk memilih objek yang akan diikuti.

Pada saat mode manual diaktifkan maka gerakan navigasi robot sepenuhnya diatur oleh pengguna. Mode mode memilih objek adalah adalah mode menggerakkan krusor untuk memilih objek untuk di ikuti. Mode mendekati objek yaitu mode mendekati objek yang telah ditentukan dengan Mode memilih objek berdasarkan pengolahan citra.

3.2 Perancangan Perangkat Keras



Gambar 3.2 Diagram perancangan perangkat keras robot

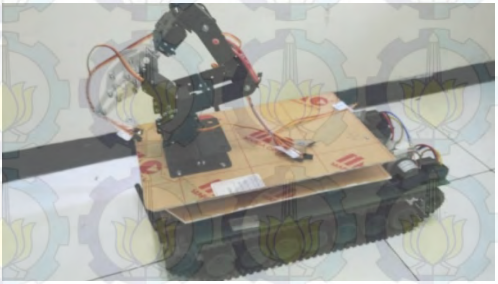
Gambar 3.2 merupakan keseluruhan perangkat keras yang terdapat dalam sistem. Perangkat yang digunakan dalam sistem ini

secara terperinci adalah sebagai berikut:

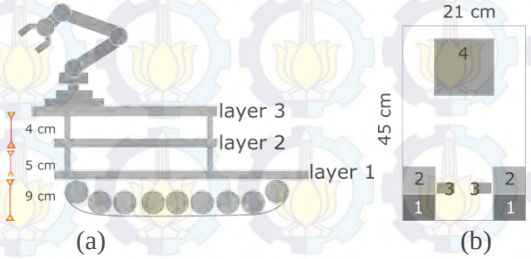
1. Laptop.
2. Tranceiver 3DR.
3. Lengan robot.
4. Video sender dan juga video receiver.
5. Motor DC.
6. ESC.
7. Ardupilot.
8. Arduino Mega.
9. Raspberry Pi
10. Kamera
11. Radio kontrol
12. Digital kompas dan GPS
13. Remot kontrol.
14. Receiver

Setiap bagian perangkat keras, kegunaan dan spesifikasinya akan dijelaskan secara lebih terperinci pada bagian berikutnya

3.2.1 Desain Robot



Gambar 3.3 Foto robot



(a) (b)

Gambar 3.4 Ilustrasi ukuran robot. (a) tampak samping (b) tampak dari layer 3

(a) (b)

Gambar 3.5 Ilustrasi ukuran robot. (a) tampak dari layer 1 (b) tampak dari layer 2

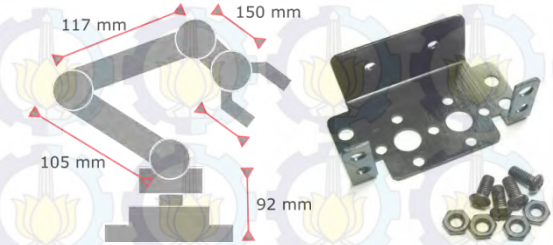
Keterangan ilustrasi robot:

1. ESC
2. Rotary encoder
3. Motor DC
4. Lengan robot
5. Baterai
6. Video sender (pengirim sinyal video)
7. Raspberry Pi
8. Multiplexer analog
9. Regulator Baterai
10. Arduino Uno
11. Arduino Mega
12. 3DR telemetry
13. GPS dan Kompas digital

Moblie robot yang dirancang memiliki bentuk dasar menyerupai tank. robot digerakkan menggunakan dua motor dc. Robot di desain berlapis untuk mempermudah peletakan komponen. Layer 1 adalah lapisan robot yang paling bawah. Lengan robot ditempatkan pada layer 3 dan pada posisi robot yang paling depan, sehingga mempermudah pergerakan lengan. Kamera diletakkan pada lengan dan juga pada layer 2. Kamera pada layer 2 adalah kamera navigasi. Kamera yang lainnya diletakkan pada lengan dikarenakan kamera tersebut digunakan untuk mendetekdi objek.

Lengan robot terdiri dari 5DOF. Sendi (*joint*) lengan robot terbuat dari motor servo TowerPro MG996R .Lengan robot terbuat

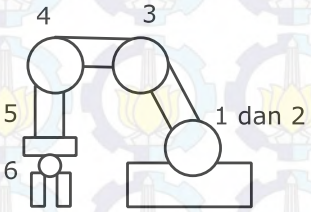
dari bracket standard servo. Berikut ini adalah rincian ukuran lengan robot.



(a) (b)

Gambar 3.6 (a)Rincian ukuran lengan robot (b) Multi purpose servo bracket

Lengan robot terdiri dari enam motor servo. Keterangan peletakan posisi motor servo dapat dilihat pada Gambar 3.7 dibawah. Motor servo pertama dan kedua adalah duabuah motor servo yang dirangkap. Motor servo pertama dan kedua dirangkap untuk menambah daya motor servo sehingga motor servo dapat mengangkat beban yang lebih besar. Motor servo yang ketiga dan keempat adalah motor servo bisa yang terangkai secara seri. Gerakan motor servo pertama, kedua, ketiga, dan keempat diatur menggunakan *inverse kinematics*. Motor servo berikutnya adalah motor sevo yang kelima. Motor servo yang kelima berfungsi sebagai pemutar orientasi capit. Orientasi capit pada tugas akhir ini sengaja dibuat dapat diputar supaya pengguna dapat menyesuaikan orientasi capit terhadap orientasi benda. Motor servo keenam adalah motor servo untuk menggerakkan capit. Capit pada motor servo hanya digerakkan dengan mode membuka dan menutup.

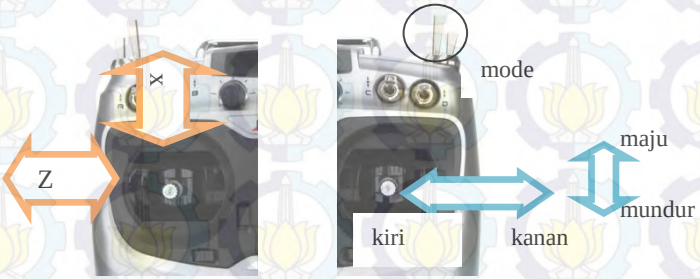


Gambar 3.7 Peletakan enam buah motor servo

3.2.2 Pengontrol Radio dan Penerima Sinyal Radio

Gambar 3.8 Pengontrol radio (kiri) penerima sinyal radio (kanan)

Pengontrol radio menggunakan RadioLink AT9, sedangkan receiver menggunakan RD9 RadioLink. Radio kontrol digunakan sebagai antarmuka pengguna untuk menavigasikan robot. RadioLink AT9 adalah pengirim sinyal radio sinyal sedangkan *receiver* RD9 adalah penerima sinyal radio dari RadioLinkAT9. Penerimasinyal radio ini dapat menerima sinyal dari 9 kanal, namun pada penelitian ini hanya akan digunakan 9 kanal.



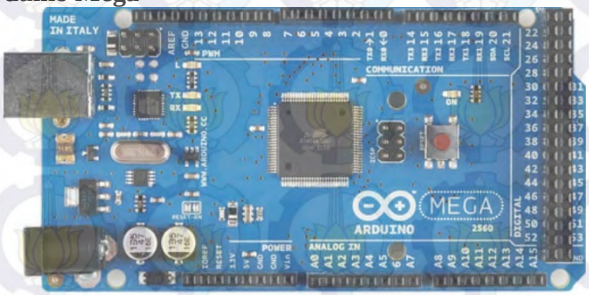
Gambar 3.9 Ilustrasi pengendalian lengan (kiri) dan ilustrasi pngendalian gerak robot (kanan)

Gambar 3.9 adalah penjelasan kontrol yang digunakan dalam penelitian ini. Remot kontrol digunakan untuk mengontrol navigasi robot, lengan robot, dan juga sebagai *joystick*(tuas) untuk memilih benda yang akan didekati melalui kamera.Pada remot kontrol terdapat saklar yang digunakan untuk mengganti mode diantara mode manual, mode otomatis mengikuti titik, dan mode otomatis pulang.

Penerima sinyal radio digunakan untuk menerjemahkan membaca sinyal yang dikirim oleh remot. Sinyal pertama yang dibaca adalah sinyal PWM dari salah satu kanal pada remot yang kemudian

diterjemahkan menjadi pemilihan mode navigasi. Kemudian sinyal kedua adalah sinyal PWM dari kanal yang kemudian diterjemahkan menjadi perintah menggerakkan lengan menggunakan *inverse kinematics*. Ketiga adalah sinyal PWM dari dua kanal untuk menggerakkan motor robot. Kemudian sinyal PWM dari satu kanal untuk menggerakkan kamera. Penjelasan mengenai software akan dijelaskan pada bagian Perancangan software. Penjelasan mengenai pengkabelan antar hardware akan dijelaskan pada bagian hardware berikutnya.

3.2.3 Arduino Mega



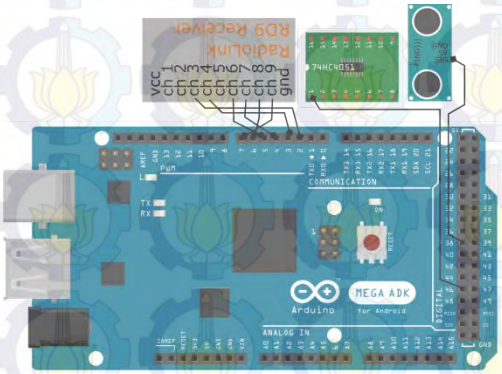
Gambar 3.10 Arduino Mega

Arduino Mega digunakan utntuk mengontrol gerakan lengan robot. Pada arduino mega terdapat program inverse kinematic lengan, sehingga dapat memudahkan pengguna dalam mengontrol gerakan lengan. Arduino mega mengontrol gerakan lengan dengan cara membaca data dari receiver kemudian diolah dengan inverse kinematic.

Berikut ini adalah spesifikasi singkat Arduino Mega

Microcontroller	ATmega1280
Operating Voltage	5V
Input Voltage (recommended)	7-12V
Input Voltage (limits)	6-20V
	Digital I/O Pins 54 (15 diantaranya menghasilkan output PWM)
Analog Input Pins	16
DC Current per I/O Pin	40 mA
DC Current for 3.3V Pin	50 mA

Flash Memory	128 KB (4 KB dari memori digunakan sebagai bootloader)
SRAM	8 KB
EEPROM	4 KB
Clock Speed	16 MHz

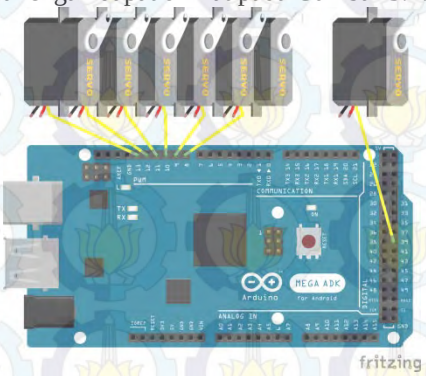


Gambar 3.11 Pengkabelan Arduino Mega dengan receiver, multiplexer dan sensor ultrasonik

Arduino dihubungkan dengan sensor ultrasonik sebagai bengukur jarak. Sensor ultrasonik digunakan sebagai pengaman robot supaya tidak menabrak. Pada saat mode mendekati objek, sensor ultrasonik digunakan untuk memberhentikan robot ketika sudah berada pada jarak tertentu terhadap benda. Kemudian terdapat IC tambahan yang terhubung dengan Arduino Mega, yaitu IC 4051. IC 4051 adalah *multiplexer* analog. *Multiplexer* pada tugas akhir ini digunakan untuk memilih video yang akan ditampilkan dilayar. Arduino Mega dan *multiplexer* menggunakan komunikasi satu bit untuk memilih dua input kamera. Keterangan Mengenai mode akan dijelaskan pada bagian perancangan software. Keterangan pengkabelan Arduino Mega dengan Arduino Uno akan dijelaskan pada bagian Arduino Uno

Pada Gambar 3.12terdapat enam pin dari Arduino Mega dengan enam pin dari penerima sinyal radio yang berarti enam kanal PWM dari penerima sinyal radio. Arduino mega membaca satu kanal (Kanal6) untuk pemilihan mode navigasi otomatis mendekati objek atau mode navigasi manual. Kemudian terdapat dua kanal (Kanal 3 dan Kanal 7) dari receiver untuk menggerakkan lengan. Dua kanal penerima sinyal radio diartikan menjadi gerakan maju dan kebawah

oleh lengan melalui *inverse kinematics.* Satu kanal berikutnya (Kanal 9) adalah untuk menggerakkan capit lengan. Kemudian kanal berikutnya (Kanal 8) untuk memutar ujung lengan. Ujung lengan dapat diputar untuk memudahkan pengguna dalam mengambil objek dengan orientasi berbeda. Kanal berikutnya (Kanal 4) digunakan untuk menggerakkan servo pada kamera. Motor servo dipasang pada kamera untuk memudahkan pengguna dalam menavigasikan robot. Pada Gambar 3.12 berikut ini adalah ilustrasi pengkabelan antara Arduino Mega dengan motor servo pada lengan robot.. Keterangan penomeran motor servo pada lengan dapat dilihat pada Gambar 3.7.



Gambar 3.12 Pengkabelan Arduino Mega dengan motor servo pada lengan dan kamera.

3.2.4 Arduino Uno

Pada Tugas akhir ini Arduino Uno digunakan untuk mengontrol laju kecepatan robot. Pengendalian laju kecepatan robot tidak dibuat satu sistem dengan Arduino Mega karena dapat mengganggu performa Arduino Mega. Spesifikasi Arduino Uno yang digunakan adalah sebagai berikut:

Mikrokontroler	ATmega328P
Tegangan operasi	5V
Tegangan input	7-12V
Input Voltage (limit)	6-20V
Pin I/O	14 (6 diantara menghasilkan dapat PWM)
Arus DC pada pin I/O	20 mA
Flash Memory	32 KB, 0.5 KB digunakan sebagai bootloader

SRAM 2 KB
EEPROM 1 KB
Frekuensi Clock 16 MHz

Arduino uno juga digunakan untuk menerjemahkan output PWM dari Ardupilot. Arduino Uno menggunakan PWM dari Ardupilot untuk kemudian diteruskan kemotor apabila pengguna memilih untuk mengendalikan robot secara manual. Arduino meneruskan PWM dari Ardupilot ke motor driver atau ESC. Keterangan pengkabelan Arduino Uno dan ESC akan dijelaskan pada bagian ESC

Arduino Uno juga digunakan untuk berkomunikasi dengan Raspberry Pi. Data yang dikomunikasikan antara Raspberry Pi menuju Arduino Uno adalah perintah untuk menggerakkan robot oleh Raspberry Pi. Sedangkan perintah yang dikirim dari arduino Uno menuju Raspberi Pi adalah perintah mengendalikan kursor Raspberry Pi melalui remot. Kursor digerakkan seolah-olah pengguna menggunakan *joystick*. Kursormerupakan fitur yang sangat penting karena kursor pada Raspberry Pi digunakan untuk memilih benda yang akan didekati secara otomatis.

Arduino Uno berkomunikasi dengan Arduino Mega untuk memilih mode. Arduino Uno tidak melakukan pembacaan mode dengan sendirinya melaikan melalui Arduino Mega. Pembacaan pemilihan mode dilakukan oleh Arduino Mega melalui pembacaan sinyal PWM dari penerima sinyal radio kanal ke 5. Pembacaan sinyal PWM yang dilakukan Arduino Mega dapat mengurangi performa Arduino Uno apabila dilakukan Arduino Uno.

3.2.5 Ardupilot, Digital Kompas, GPS , 3DR Telemetry

Digital kompas dan GPS adalah fitur utama yang digunakan dalam gerakan robot secara otomatis melalui koordinat GPS. Digital kompas digunakan untuk mengetahui orientasi robot terhadap arah mata angin. GPS digunakan untuk mengetahui posisi kooordinat robot. Dalam pengerjaan sistem pada tugas akhir ini data dibaca oleh Ardupilot.

3DR telemetry digunakan untuk mengirimkan data status robot. Data yang dikirimkan adalah data kemiringan, ketinggian, kecepatan, kompas, dan koordinat. Data-data tersebut sangat berguna untuk

menavigasikan robot melalui jarak jauh. Data tersebut dapat dilihat di computer menggunakan software yang disediakan Ardupilot.

Ardupilot digunakan untuk mengontrol gerakan robot secara otomatis. Ardupilot membaca data dari kompas dan GPS. Untuk melakukan gerakan otomatis, Ardupilot membaca dan membandingkan data koordinat yang diinginkan dan data koordinat robot. Ardupilot mengurangi error atau selisih antara data data koordinat yang diinginkan dan data koordinat robot dengan cara bergerak menuju koordinat yang diinginkan. Robot bergerak maju untuk menuju titik yang diinginkan, meskipun titik tersebut berada di belakang robot. Sehingga robot harus bergerak berputar untuk menuju koordinat yang berada di belakang robot. Robot dapat mengetahui orientasi arah mata angin dengan kompas.

Pada Ardupilot juga terdapat gyroscope dan accelerometer. Gyroscope adalah sensor untuk mengetahui kemiringan sedangkan accelerometer adalah sensor untuk mengetahui percepatan benda. Pada Ardupilot juga terdapat pressure sensor. Barometer atau pressure sensor digunakan untuk mengetahui ketinggian benda. Sensor-sensor ini digunakan untuk mengetahui status kemiringan, orientasi mata angin, percepatan dan juga ketinggian robot. Status ini digunakan untuk memonitor robot sehingga dapat mempermudah pengguna untuk mengendalikan robot dari jarak jauh.

Ardupilot, kompas, GPS, dan telemetri pada Tugas akhir ini merupakan satu kesatuan sistem. Ardupilot digunakan unutk memudahkan pengguna dalam navigasi jarak jauh. Ardupilot mengolah seluruh data dari GPS dan kompas menjadi dua sinyal PWM untuk menggerakkan dua motor. Pada tugas akhir ini output PWM pada Ardupilot tidak langsung menggerakkan motor. Sinyal output PWM dari ardupilot diolah kembali oleh Adruino UNO

3.2.6 Kamera dan Multiplexer

Pada sistem ini terdapat dua buah kamera. Kamera pertama adalah kamera video analog buatan AOMWAY. Kamera ini juga biasa disebut kamera FPV. Kemudian kamera kedua adalah kamera webcam Logitech c170. Kamera adalah alat bantu yang sangat penting dalam navigasi jarak jauh atau navigasi tanpa awak. Dua kamera ini memiliki fungsi yang berbeda, kamera FPV digunakan untuk memudahkan navigasi sedangkan kamera webcam digunakan untuk pengolahan citra. Pada Tugas akhir ini pengguna hanya dapat melihat gambar dari

satu kamera. Gambar dari kamera dikirim ke monitor melalui video sender.

Berikut ini adalah spesifikasi singkat AOMWAY mini CMOS camera:

Image sensor	CMOS
TV System	PAL/NTSC
Horizontal resolution	600TVL
Picture elements	NTSC:510x492(H), PAL: 500x582(V)
S/N Ratio	>48 dB
Y Characteristic	0.45
Usable illumination	0.1 Lux
Sync system	Internal synchronization
Electronic shutter time	Auto, NTSC :1/60 – 1/100000 Sec PAL : 1/60 – 1/100000 Sec
Scanning system	NTSC : 525 Lines, 60 Field/sec PAL : 625 Lines, 50 Field/sec
Video output	1 Vp-p75ohm
Operational temp	$-20^{O}C$ – $65^{O}C$ RH95%MAX
Power supply	DC 3.6 – 5.5v max 80mA

Sebelum sinyal gambar dari kamera dikirimkan kemonitor melalui video sender, sinyal gambar terlebih dahulu dipilih melalui multiplexer analog, yaitu IC 4051. IC 4051 memiliki delapan input dan satu output. Tugas akhir ini hanya menggunakan dua input dari IC 4051 untuk dikeluarkan salah satu menuju video senderSelector dari IC 4051 di dikontrol menggunakan Arduino Mega. User dapat memilih gambar yang mana yang akan ditampilkan dari dua kamera menggunakan remot kontroler secara manual. Untuk memilih output IC 4051 menggunakan selector.

3.2.7 Raspberry Pi

Raspberry Pi digunakan untuk pengolahan citra. Raspberry Pi melalakukan pengolahan citra dengan menggunakan *lucas kanade* pada OpenCV. OpenCV adalah *library* pemrograman yang berisi berbagai fungsi pengolahan citra. Sedangkan *lucas kanade* adalah suatu metode *object tracking* (penjajakan object) pada pengolahan citra.

Pada umumnya pengolahan citra dilakukan menggunakan komputer atau laptop. Laptop lebih cocok untuk diaplikasikan pada robot mobil daripada komputer. Akan tetapi pada robot mobil yang kecil dibutuhkan komputer yang lebih kecil, sehingga Raspberry Pi cocok untuk digunakan pada sistem ini. Berikut ini adalah spesifikasi Raspberry Pi:

Cip	Broadcom BCM2836 SoC
Arsitektur	Quad-core ARM Cortex-A7
CPU	900MHz
GPU	Dual Core VideoCore IV® Multimedia Co-Processor
Memori	1GB LPDDR2
Sistem operasi	Linux Raspbian
Dimensi	85 x 56 x 17mm
Keluaran video	HDMI atau RCA
GPIO	27 Pin dengan tegangan 3.3 V

Raspberry Pi berkomunikasi serial dengan Arduino Uno. Komunikasi dari Arduino berisi perintah untuk menggerakkan pointer saat memilih object untuk di dekati. Komunikasi dari Raspberry Pi menuju Arduino Uno berisi perintah untuk menggerakkan robot kekanan dan kekiri untuk mengikuti objek yang telah melalui pengolahan citra.

3.2.8 *Video Sender, Video Receiver* dan *Monitor Kamera*

Video sender(pengirim sinyal video) adalah perangkat untuk mengirimkan data video analog. *Video receiver*(penerima sinyal video) adalah perangkat untuk menerima sinyal video analog. Data video analog tersebut kemudian ditampilkan di monitor. Data video berasal dari dua sumber yang berbeda yaitu Raspberry Pi dan juga kamera

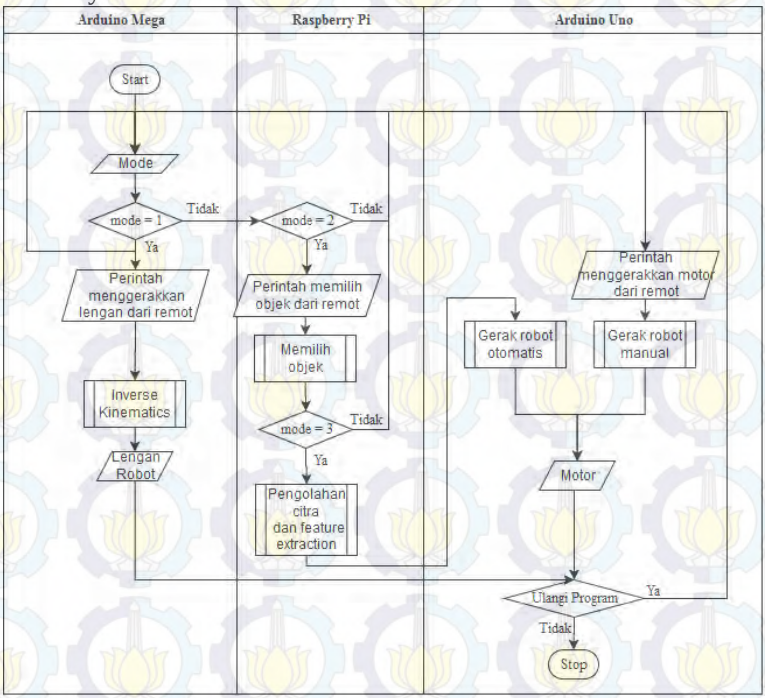
analog. Dua sinyal ini dipilih menggunakan multiplexer, kemudian dikirim menggunakan pengirim sinyal video.

3.2.9 ESC dan DC motor

ESC adalah singkatan dari*Electric Speed Control*. ESC digunakan untuk menggerakkan motor DC. Pada ESC terdapat regulator arus dan tegangan sehingga dapat mengatur penggunaan daya dan kecepatan motor DC. ESC dikontrol menggunakan PWM. PWM yang mengontrol kecepatan motor DC di keluarkan oleh Arduino Uno Pada tugas akhir ini terdapat dua motor DC dan dua ESC. Dua motor pada tugas akhir ini memiliki peran untuk menggerakkan robot baik maju maupun berbelok. Pada komunitas robot, cara mengontrol navigasi seperti ini lebih dikenal dengan *differential steer*. yaitu mengontrol gerak robot melalui perbedaan kecepatan roda

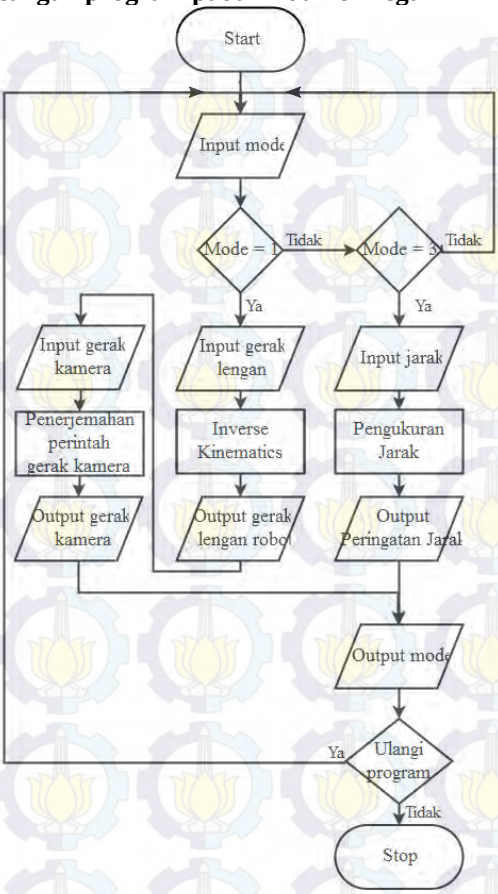
3.3 Perancangan Perangkat Lunak

Pada bagian ini akan dijelaskan mengenai penyusunan pernagkat lunak. Perangkat lunak pada tugas akhir ini dibagi menjadi tiga bagian atau unit proses. Unit proses pertama adalah Raspberry Pi. Unit proses ini digunakan untuk pengolahan citra. Unit proses yang kedua adalah adalah Arduino Mega. Pada unit proses ini terdapat proses penggerak lengan robot. Unit proses yang ketiga adalah Arduino Uno. Pada unit proses ini terdapat proses penggerak motor robot. Ketika robot mode otomatis mengikuti objek, gerakan motor robot memiliki umpan balik terhadap kamera. Gambar dibawah ini adalah diagram alir dari keseluruhan program. Berikut ini adalah gambaran keseluruhan kerja sistem. Untuk penjelasan perangkat lunak lebih detail pada setiap unit prosesnya akan dijelaskan pada bagian berikutnya.



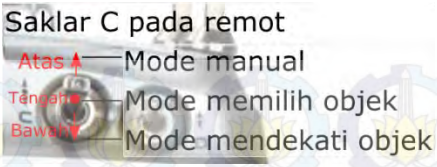
Gambar 3.13 Diagram alir keseluruhan program

3.3.1 Perancangan program pada Arduino Mega



Gambar 3.14 Diagram alir program pada Arduino mega

Gambar 3.14 adalah diagram alir keseluruhan program pada Arduino Mega. Arduino Mega memiliki empat input dan empat output. Input yang pertama pada Arduino Mega adalah pemilihan mode. Pemilihan mode pada Arduino Mega dilakukan menggunakan remot. Pemilihan mode dilakukan dengan saklar C pada remot. Setiap perintah yang dikirim dari remot diterima oleh receiver. Arduino Mega membaca sinyal PWM dari receiver.



Gambar 3.15 Ilustrasi pemilihan mode menggunakan saklar c pada remot.

Pada Arduino, pembacaaan sinyal PWM menggunakan fungsi yang telah disediakan oleh Arduino, yaitu *void pulseIn(void)* .PWM yang dibaca oleh Arduino memiliki kisaran*duty cycle*6,5% sampai dengan 12%. Dengan memanfaatkan kisaran tersebut dibuatlah suatu kondisi pemilihan mode ataupun perintah yang lain. Pemilihan mode menghasilkan menghasilkan output data yang dikirim ke Arduino Uno. Data yang dikirim dari Arduino Mega menuju Arduino Uno adalah data dua bit. Tabel mengenai isi data tersebut dapat dilihat pada tabel dibawah. Informasi data ini dikirim ke Arduino Uno menggunakan pin GPIO.

Mode	**Kode**	**Pin 44**	**Pin 45**
Mode Manual	Mode = 1	LOW	HIGH
Mode Memilih Objek	Mode = 2	HIGH	LOW
Mode Mendekati Objek	Mode = 3	HIGH	HIGH

Tabel 3.1 Pemilihan mode dan pengiriman data mode ke Arduino Uno

Input yang kedua yang digunakan poda Arduino Mega adalah input dari remote untuk menngerakkan kamera. Kamera pada tugas akhir in digerakkan menggunakan motor servo, yaitu sebagai output dari proses menggerakkan kamera. Pembacaan perintah menggerakkan kamera ini memiliki cara yang sama dengan pembacaan input mode. Gerakan kamera ini menggunakan tuas kiri pada kamera. Ilustrasi gerakan motor servo kamera dan input dari remot dapat dilihat pada gambar dibawah ini.

Gambar 3.16 Ilustrasi gerkan kamera (kanan) diiringi dengan input dari remot (kiri).

Input yang ketiga dan keempat adalah gerakan memutar capit dan memcapit. Gerakan memutar capit sangat penting karena diperlukan untuk menyesuaikan capit terhadap orientasi benda. Capit dapat diputar dengan pertambahan sekitar 1^{o}. Kemudian gerakan mencapit diperlukan untuk menyesuaikan capit dengan benda. Gerakan memutar capit digerakkan menggunakan potensiometer 1 pada remot. Pada gerakan mencapit ini hanya digunakan dua gerakan, yaitu membuka dan menutup. Gerakan mencapit di perintahkan melalui saklar D pada remot. Ilustrasi gerakan mencapit, dan memutar capit terhadap input dapat dilihat pada gambar dibawah. Perancangan gerakan lengan menggunakan *inverse kinematics* akan dijelaskan pada bagian berikutnya.

Gambar 3.17 Ilustrasi capit menutup (kiri) dan capit membuka (kanan)

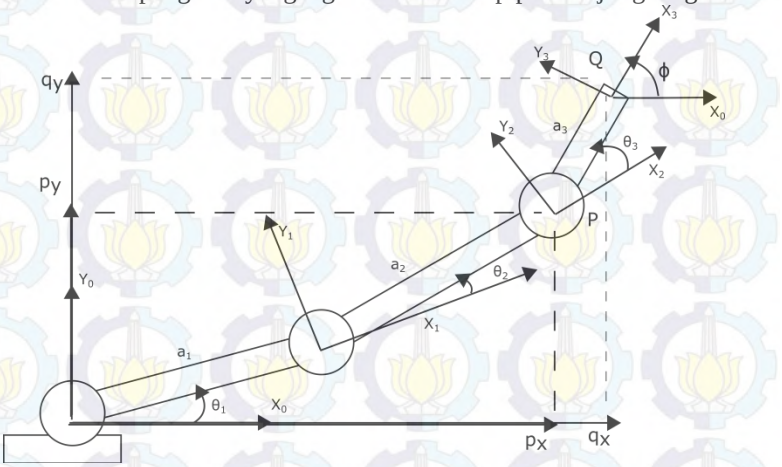
Input yang kelima adalah input jarak. Input jarak diambil dari sensor ultrasonik. Pengukuran jarak untrasonic dihitung berdasarkan cepat rambat suara pada udara dan waktu yang ditempuh. Penjelasan mengenai perhitungan yang digunakan untuk mengukur jarak menggunakan sensor ultrasonik terdapat pada bagian dasar teori. Pada tugas akhir ini jarak yang didapatkan dari pengukur jarak digunakan sebagai peringatan bahwa benda sudah berada pada jarak tertentu, dalam hal ini digunakan jarak 20cm. Output peringatan diumpankan pada unit proses Arduino Uno. Komunikasi yang digunakan untuk menghantarkan data ini hanyalah berupa data satu bit sehingga komunikasi ini hanya menggunakan GPIO. Keterangan lebih lanjut mengenai kontrol jarak akan dijelaskan pada bagian berikutnya. Berikut ini adalah tabel data yang dikirim.

Input Jarak	**Kode**	**Pin 43**
Jarak kurang dari 20 cm	Ping_ok = 0	LOW
Jarak lebih dari 20 cm	Ping_ok = 1	HIGH

Tabel 3.2 Pengiriman data peringatan jarak ke Arduino Uno

3.3.1.1 Perancangan gerak lengan robot dengan inverse kinematics

Input yang keenam pada Arduino Mega adalah input untuk menggerakkan lengan yang diproses menggunakan Inverse Kinematics. Pada lengan robot tugas akhir ini digunakan lengan 4 DOF. Ilustrasi gambar lengan 4 DOF yang digunakan pada tugas akhir ini dapat dilihat dibawah. Pada gambar tersebut terdapat empat sudut, tiga sudut relative terhadap poros pusat dan satu sudut relative terhadap poros utama. Pada gambar tersebut setiap sudut lengan berotasi searah atau berlawanan dengan jarum jam. Sehingga pada tugas akhir ini setiap sudut lengan robot tidak memiliki sudut putar terhadap sumbu *x* atau *twist*.Namun pada tugas akhir ini sudut putar yang digunakan pada setiap sudut lengan robot adalah mulai dari 0^o sapmai dengan -90^o. Pada gambar tersebut terdapat pula suatu titik yang diberinama titik Q. Titik Q pada gambar tersebut adalah *end effector* atau ujung lengan. Titik Q lengan robot adalah capit pada konstruksi hardware lengan. Pada inverse kinematics ini tidak membahas putaran capit dan juga gerakan mencapit karena tidak memberikan pengaruh yang signifikan terhadap posisi ujung lengan.



Gambar 3.18 Lengan 4 DOF

Sebelum membahas *inverse kinematics* pada lengan robot, terlebih dahulu harus ditemukan *forward kinematics* dari lengan robot. *Forward kinematics* lengan robot tertera pada rumus dibawah ini:

$$q_x = a_1 \cos\theta_1 + a_2 \cos(\theta_1 + \theta_2) + a_3 \cos(\theta_1 + \theta_2 + \theta_3) \quad (5)$$

$$q_y = a_1 \sin\theta_1 + a_2 \sin(\theta_1 + \theta_2) + a_3 \sin(\theta_1 + \theta_2 + \theta_3) \quad (6)$$

$$\phi = \theta_1 + \theta_2 + \theta_3 \quad (7)$$

$$p_x = q_x - a_3 \cos\phi \quad (8)$$

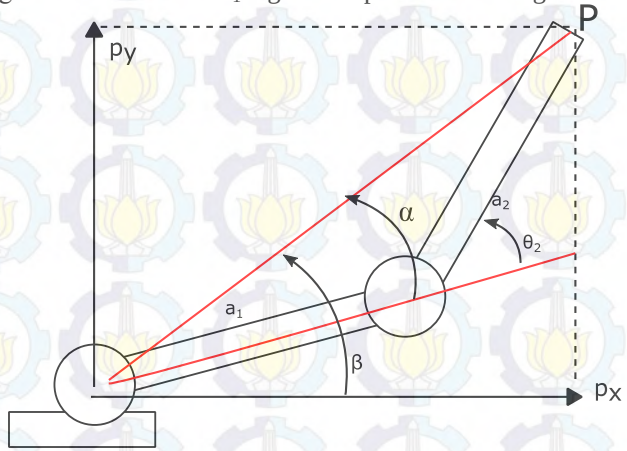
$$p_y = q_y - a_3 \sin\phi \quad (9)$$

Forward kinematics adalah rumus yang digunakan untuk mengetahui posisi ujung terhadap sumbu *x* dan sumbu *y* dengan input sudut tertentu. Sedangkan *inverse kinematics* adalah rumus yang digunakan untuk mengetahui setiap sudut dengan input posisi ujung terhadap sumbu *x* dan sumbu *y*. Sebelum menuju *Inverse kinematik*, maka terlebih dahulu harus diketahui setiap persamaan disetiap sudutnya. Dengan menggunakan aturan *cos* maka akan didapatkan θ_2 sebagai berikut:

$$p_x^2 + p_y^2 = a_1^2 + a_2^2 + 2a_1 a_2 \cos\theta_2 \quad (10)$$

$$\theta_2 = \cos^{-1}\left(\frac{p_x^2 + p_y^2 - a_1^2 - a_2^2}{2a_1 a_2}\right) \quad (11)$$

Sedangkan untuk mencari θ_1 digunakan persamaan sebagai berikut:



Gambar 3.19 Mencari sudut θ_1

$$\theta_1 = \beta - \alpha \quad (12)$$

$$\theta_1 = \tan^{-1}\left(\frac{p_y}{p_x}\right) - \tan^{-1}\left(\frac{a_2 \sin\theta_2}{a_1 + a_2 \cos\theta_2}\right) \quad (13)$$

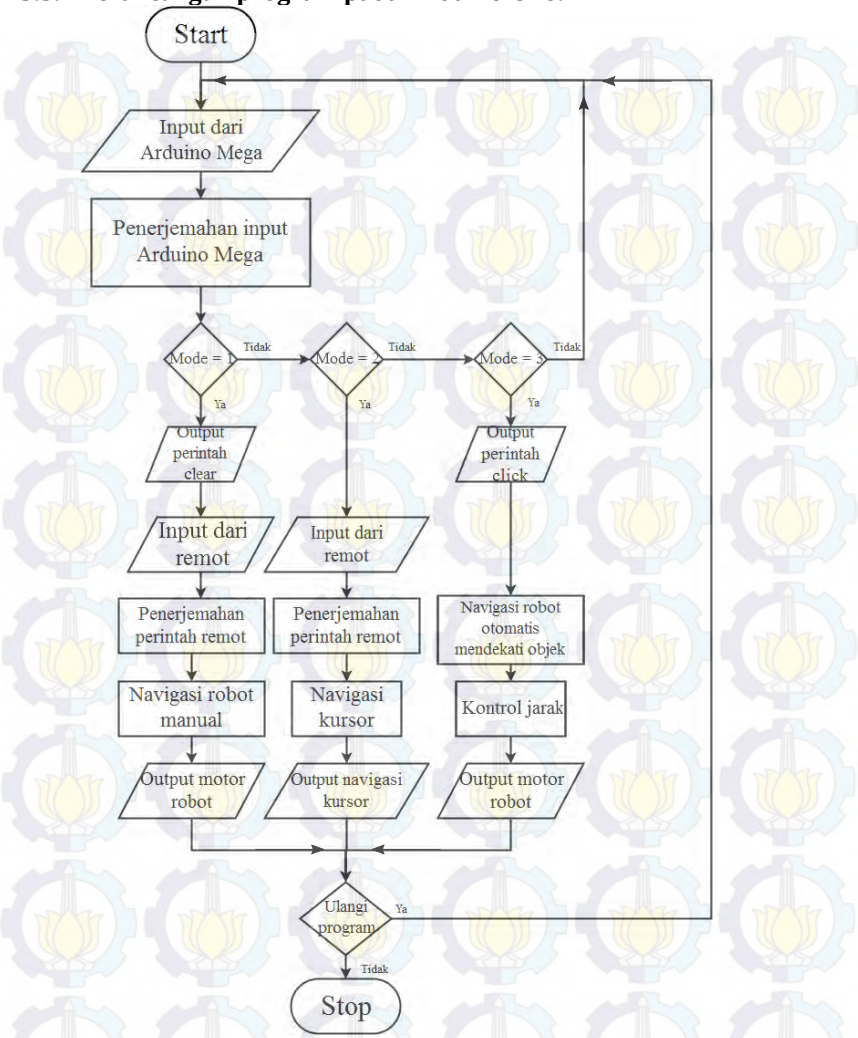
Untuk memcari θ_3 digunakan persamaan sebagai berikut:

$$\theta_3 = \phi - \theta_1 - \theta_2 \quad (14)$$

Dengan menggunakan persamaan diatas didapatkan persaaan setiap sudut lengan. Berdasarkan persaman diatas sudut dapat dihasilkan dengan memasukkan p_x, p_y, dan φ. Pada perancangan program gerak lengan robot dalam tugas akhir ini hanya digunakan dua input, yaitu p_x, p_y. Kelebihan menggunakan dua input adalah kemudahan dalam mengontrol lengan robot dan juga menyederhanakan program. Ilustrasi pergerakan lengan terhadap input dari remot dapat dilihat pada gambar dibawah ini.

Gambar 3.20 Gerakan lengan seiring dengan perintah dari remot.

3.3.2 Perancangan program pada Arduino Uno.



Gambar 3.21 Diagram alir program pada Arduino Uno

Gambar 3.21 adalah gambar diagram alir programpada Arduino Uno. Arduino Uno memiliki lima otput, namun secara hardware hanya terdapat dua output. Output motor robot memiliki hardware yang

sama. Output perintah klik, output perintah *clear*, dan juga output navigasi kursor juga memiliki hardware yang sama, yaitu unit proses Raspberry Pi. Arduino Uno memiliki empat input. Dua input pertama dapat dilihat pada diagram alir, sedangkan dua input yang kedua terdapat dalam proses "Mode navigasi robot mendekati objek". Proses "Mode navigasi robot mendekati objek" akan dijelaskan pada bagian selanjutnya sedangkan proses lainnya akan dijelaskan pada bagian ini.

Pada saat Arduino Uno pertamakali dijalankan, Arduino Uno lebih dahulu mengenali input mode dari Arduino Mega. Output Mode pada Arduino mega dapat dilihat pada Tabel 3.3. Proses penerjemahan dari Arduino Mega hanyalah operasi logika menerjemahkan perintah tersebut. Proses ini menghasilkan tiga buah mode, sama seperti pada output Arduino Mega (Tabel 3.3). Dua diantara Output tersebut memiliki perintah langsung terhadap Raspberry Pi, yaitu perintah klik dan *clear*. Perintah *click* dan *clear* dikirim ke Raspberry Pi menggunakan GPIO antara Raspberry Pi dan Arduino Uno.

Mode pertama adalah mode untuk menggerakkan robot secara manual. Dalam proses penerjemahan remot, perintah dari remot tidak langsung menuju Arduino, namun melalui receiver dan Ardupilot. Receiver menerjemahkan perintah dari remot menjadi sinyal PWM, kemudian sinyal PWM diteruskan ke Ardupilot. Didalam proses Ardupilot terdapat penerjemahan perintah remot menjadi perintah pengendalian robot dengan dua motor. Namun, output perintah dari Ardupilot dibaca oleh Arduino Uno untuk diteruskan ke motor. Hal ini dilakukan supaya Arduino Uno dapat memegang kendali motor robot.

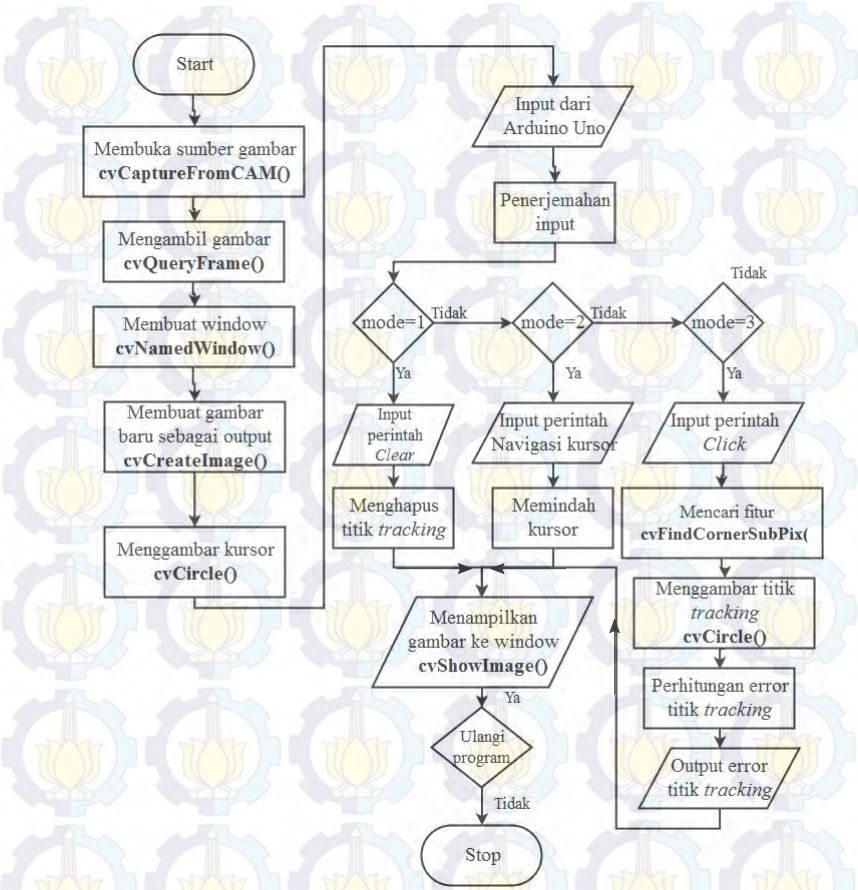
Mode kedua adalah mode penerjemahan remot menjadi perintah menvigasikan kursor. Sama seperti pada mode pertama, perintah dari remot di terjemahkan tidak langsung dari remot, namun melalui receiver dan Ardupilot. Pada proses ini Arduino Uno menerjemahkan perintah dari Ardupilot menjadi perintah menggerakkan kursor. Perintah menggerakkan kursor dikirim dari Arduino Uno ke Raspberry Pi menggunakan pin GPIO pada Arduino Uno dan Raspberry Pi. Data navigasi tersebut digabung dengan dara perintah *click* dan *clear* isi data tersebut dapat dilihat pada tabel dibawah ini.

Perintah	**Kode**	**Pin 11**	**Pin 10**	**Pin 9**
Perintah *click*	Mode 3	HIGH	LOW	HIGH
Perintah *clear*	Mode 1	HIGH	HIGH	LOW
Navigasi kursor keatas	Mode 2	LOW	HIGH	HIGH
Navigasi kursor kebawah		LOW	LOW	LOW
Navigasi kursor kekanan		LOW	LOW	HIGH
Navigasi kursor kekiri		LOW	HIGH	LOW
Navigasi diam		HIGH	LOW	LOW

Tabel 3.3 Data komunikasi Arduino Uno ke Raspberry Pi

Mode ketiga adalah mode otomatis robot mendekati objek. Didalam mode ini terdapat proses navigasi robot mendekati objek dan kontrol jarak. Proses ini akan dijelaskan pada bagian berikutnya. Dalam mode ini terdapat output perintah *click* menuju Raspberry Pi dan output motor robot.

3.3.3 Perancangan program pada Raspberry Pi dan pengolahan citra.



Gambar 3.22 Diagram alir program pada Raspberry Pi

Raspberry Pi adalah unit proses yang bertugas sebagai pengolah citra. Pada program didalam unit proses Raspberry Pi terdapat pertukaran input dan output. Raspberry Pi melakukan komunikasi data dengan Arduino Uno. Raspberry Pi menerima input perintah *click*, perintah *clear*, dan juga perintah navigasi kursor yang disingkat dalam tiga mode. Mode pada Raspberry Pi adalah hasil dari penerjemahan input yang berasal dari Arduino Uno. Data dari Arduino Uno

digolongkan menjadi tiga mode. Penerjemahan data dari Arduino Uno dilakukan dengan operasi logika biasa. Setiap mode memiliki proses yang berhubungan dengan pengolahan citra.

Berbeda dengan program pada unit proses lainnya yang diawali dengan pemilihan mode. Program pada Raspberry Pi diawali dengan berbagai macam inisialisasi untuk pengolahan citra. Inisialisasi pertama dalah menentukan sumber gambar. Pengolahan citra dapat memiliki berbagai sumber gambar, seperti kamera, foto, atau video. Pada tugas akhir ini pengolahan citra dimulai dengan proses menangkap gambar pada kamera. OpenCV memiliki fasilitas yang dapat memudahkan pengguna untuk pemilihan sumber gambar, yaitu *cvCaptureFromCAM()*.

Gambar bergerak yang dihasilkan oleh kamera adalah kumpulan gambar yang berurutan. Setiap cuplikan gambar tersebut dalam pengolahan citra biasa disebut *frame.* Untung mengambil *frame* dari kamera, OpenCV sudah menyediakan fungsi cvQueryFrame. Setelah *frame* diambil frame disimpan didalam memori.

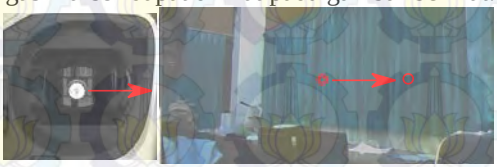
Sama Seperti program yang menghasilkan *Graphic User Interface (GUI)* atau antarmuka grafik, sebelum dapat menampilkan gambar terlebih dahulu harus dibuat jendela baru atau *window*.Fungsi yang disediakan OpenCV untuk membuat window baru adalah *cvNamedWindow()*.

Langkah selanjutnya pada program adalah membuat gambar. Gambar ini adalah gambar yang digunakan untuk output. Dalam pengolahan citra menggunakan OpenCV gambar dibuat menggunakan *cvCreateImage()*. Gambar ini adalah gambar kosong (hitam) dengan ukuran, kanal, dan kedalaman yang telah ditentukan. Setelah gambar input mengalami berbagai proses pengolahan citra, gambar akhir pengolahan dimasukkan kedalam gambar output ini.

Gambar input pada langkah selanjutnya ditambah dengan gambar lingkaran. Lingkaran ini yang nantinya akan difungsikan sebagai kursor. Ilustrasi mengenai kursor ini dapat dilihat pada Gambar 3.23. Kursor digunakan unutk memilih objek.

Gambar 3.23 Gambar kursor (lingkaran merah)

Langkah selanjutnya adalah menerjemahkan input dari Arduino Uno. Secara garis besar input Arduino Uno dibagi menjadi tiga mode. Pertama adalah mode *clear*, yaitu membersihkan gambar dati titik penjajakan. Mode kedua adalah mode navigasi kursor. Sedangkan yang ketiga adalah mode *click*. Untuk mempermudah pemahaman, terlebih dahulu akan dijelaskan mode navigasi kursor. Perintah navigasi kursor didapatkan dari Arduino Uno. Perintah navigasi kursor menggeser kursor keatas atau kebawah dan juga kekanan atau kekiri. Ilustrasi navigasi kursor dapat dilihat pada gambar berikut.

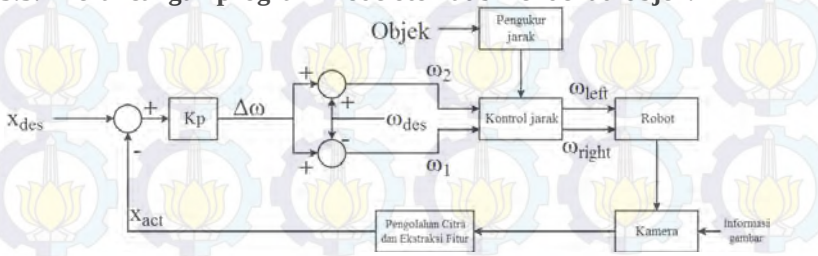


Gambar 3.24 IlustrasiKursor bergerak ke kanan (kanan) seiring dengan perintah remot (kiri).

Mode berikutnya adalah mode ketiga atau perintah *click*. Pada saat perintah ini dimasukkan, program pada Raspberry Pi melakukan dua proses, yaitu menandai titik penjajakan dan memanggil fungsi *cvFindCornerSubPix()*.Fungsi ini mencari fitur terdekat dengan kursor, yaitu berjarak 10x10 pixel. Fitur pada gambar adalah bagian gambar yang unik dan dapat ditemukan setiap urutan *frame* kamera. Fungsi ini menghasilkan titik fitur. Titik fitur ini akan terus ada mengikuti benda dimana titik itu berada, meskipun benda tersebut bergerak. Dengan demikian, titik fitur ini dapat digunakan untuk tracking. Tracking dilakukan dengan cara mengambil titik koordinat fitur. Setelah itu akan dihitung besar eror titik *tracking*.besar eror titik *tracking* dikirim sexara serial dari Raspberry Pi ke Arduino Uno.Penjelasan lebih lengkap mengenai *tracking*akan dijelaskan pada bagian Perancangan program mode otomatis mendekati objek.

Mode yang terakhir adalah mode yang pertama yaitu perinta *clear*. Pada mode ketiga atau perintah *click* telah disebutkan bahwa setelah perintah click terjadi program akan menandai titik untuk *tracking*. Pada mode ini seluruh titik fitur atau titik *tracking* dihapus. Hal ini penting dilakukan untuk menghentikan *tracking*.

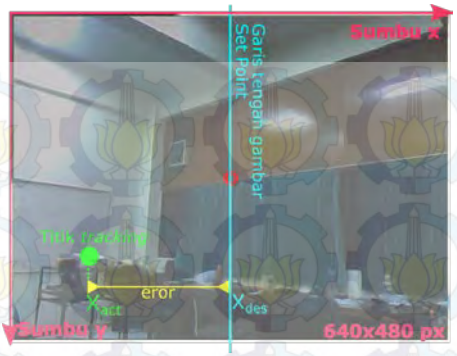
3.3.4 Perancangan program mode otomatis mendekati objek.



Gambar 3.25 Blok diagram kontrol mode otomatis mendekati objek

Metode kontrol *tracking* yang digunakan pada tugas akhir ini adalah kontrol proporsional. Kontrol yang digunakan pada sistem ini adalah kontrol haluan robot dengan tujuan suapaya haluan robot selalu menghadap target (benda yang akan didekati secara otomatis). Pada kontrol ini dilakukan dengan cara menambah atau mengurangi kecepatan motor kiri atau motor kanan robot (*differential steering*). Kontrol kecepatan pada sistem ini tidak dipengaruhi jarak target terhadap robot.

Kontrol ini memanfaatkan fitur yang di ekstrak dari pengolahan citra. Seperti yang sudah dijelaskan pada bagian sebelumnya, setalah perintah *click* di panggil maka ankan dicari titik fitur menggunakan *cvFindCornerPix()*. Setiap titik fitur memiliki koordinat, yaitu koordinat (x,y). Satu fitur yang diambil dari koordinat titik ini adalah koordinat terhadap sumbu x. Titik ini disebut dengan x_{act} atau koordinat aktual fitur terhadap sumbu x. Setelah koordinat didapatkan maka dapat dihitung eror titik penjajakan. Eror titik penjajakan adalah perbedaan antara koordinat aktual fitur terhadap sumbu x (x_{act}) dengan garis tengah gambar (x_{des}). Tujuan dari kontrol ini adalah memperkecil eror. Tujuan tersebut dapat dicapai dengan menggerakkan robot kekanan atau kekiri. Ilustrasi perhitungan eror titik penjajakan dapat dilihat pada Gambar 3.26.



Gambar 3.26 Ilustrasi perhitungan eror titik *tracking*(penjajakan).

Setelah ditemukan besar nilai eror titik penjajakan, nilai dikirim keArduino Uno mengunakan komunikasi seial. Dengan menggunakan konstanta proporsional, Arduino Uno mengubah nilai eror titik penjajakanmenjadi kecepatan yang disebut Δω. Konstanta proporsional didapatkan melalui percobaan *(trial and eror)*. Δω digunakan untuk menentukan arah robot yaitu kekanan atau kekiri. Apabila titik penjajakanberada dikiri seperti pada Gambar 3.26, maka robot di belokkan kekiri, sehingga titik akan berangsur-angsur ketengah. Membelokkan robot kekiri dilakukan dengan cara menambah kecepatan motor kanan, dan mengurangi kecepatan motor kiri. Demikan juga sebaliknya apabila titik penjajakan berada dikanan. Kemudian robot akan berhenti ketika sudah mendekati objek dengan jarak tertentu.

Kontrol jarak dilakukan dengan pengukur jarak, yaitu sensor ultrasonik. Perhitungan jarak dilakukan menggunakan Arduino Mega. Arduino Mega member peringatan setiap kali benda sudah berjarak kurang dari 20cm. Apabila Arduino Uno menerima sinyal peringatan bahwa benda berjarak kurang dari 20cm, maka Arduino Uno memberhentikan kedua motor.

Halaman Ini Sengaja Dikosongkan

BAB IV

PENGUJIAN DAN ANALISIS

Bab ini menjelaskan pengujian sistem sehingga dapat diketahui kinerjanya. Pada bab ini akan dibahas pengujian kecepatan robot untuk menghadapkan haluan ke objek, pengujian kecepatan robot mendekati objek dan osilasi robot dalam mendekati objek.

4.1 Pengujian kemampuan lengan mengangkat beban

Pengujian ini dilakukan untuk mengukur berat beban yang dapat terangkat oleh lengan. Pengujian dilakukan dengan cara menambah berat suatu beban secara bertahap, kemudian mengukur tinggi beban dapat terangkat oleh lengan. Tinggi beban terangkat diukur dari base lengan. Ilustrasi dapat dilhat pada Gambar 4.1

Pada pengujian ini diperoleh beban maksimal yang dapat diangkat oleh lengan adalah 0,2178 Kg atau 217 gram. Berikut adalah gambar hasil pengujian beban.

4.2 Pengujian tingkat keberhasilan robot dalam mendekati benda secara otomatis

Pengujian ini dilakukan untuk melihat kemampuan robot dalam mendekati benda secara otomatis. Pengujian ini dilakukan untuk mengetahui batas maksimal jarak robot dan benda sehingga robot masih dapat mendekati benda secara otomatis. Pengujian ini dilakukan dengan cara meletakkan robot dengan jarak tertentu dari objek. Kemudian robot diperintahkan untuk mendekati objek tersebut apabila robot tidak dapat mendekati objek, maka robot dinyatakan gagal. Benda yang akan didekati robot dapat dilihat pada Gambar 4.3.

Tabel 4.1 adalah hasil pengujian robot mendekati benda secara otomatis dengan jarak tertentu. Dengan menggunakan sistem yang telah dirancang dan target yang telah ditentukan robot bekerja secara optimal hingga jarak 330cm. Kemampuan robot mendekati objek secara otomatis berangsur-angsur turun dengan jarak melebihi 330cm.

4.3Pengujian kecepatan robot menghadapkan haluan ke objek

Pengujian pada bagian ini adalah pengujian untuk melihat respon pergerakan kekanan dan kekiri robot dalam mendekati objek. Pengujuan ini berhubungan dengan kontrol proporsinal robot. Pengujian ini juga berhubungan dengan ketepatan robot dalam mendekati objek. Pengujian ini dilakukan dengan cara menghitung eror antara titik yang diinginkan (yaitu titik *tracking*) dengan titik yang telah dicapai robot dalam satuan pixel.

Pengujian ini dilakukan dengan cara meletakkan robot pada jarak tertentu dengan objek, yaitu 301 meter. Kemudian benda diletakkan dengan sudut tertentu dari robot. Ilustrasi pengujian dapat dilihat di Gambar 4.4. Benda diletakkan dalam posisi tertentu sehingga pada kamera benda terlihat pada posisi sumbu tertentu seperti pada Gambar 3.26. Robot dan benda juga ditempatkan pada ruangan dengan pencahayaan tetap. Hal ini dilakukan untuk meminimalisasi pengaruh pencahayaan dan perubahan latar belakang (*background*). Percobaan dilakukan sebanyak setiap sepuluh kali dengan posisi benda yang samakemudian posisi benda di geser.

Dalam pengujian ini didapatkan tiga jenis data, yaitu *rising time*, dan *overshoot*. *Rise time* adalah waktu yang dibutuhkan saat pertamakali *tracking* dilakukan hingga mencapai titik yang diinginkan. *Rise time* bisa dianalogikan sebagai kecepatan robot menghadapkan haluan ke benda saat pertamakali titik *tracking* didapatkan. *Overshoot* adalah eror terbesar dalam *tracking. Oversoot* biasanya merupakan error pertamakali dalam *tracking*. Pada pengujian ini diambil data pengaruh posisi benda terhadap kecepatan *rise time* dan *overshoot*. Berikut ini adalah data *rise time* yang didapatkan

Seperti yang terlihat pada Gambar 4.5 bahwa waktu *rise time* berkisar diantara 12 detik .Pada data tersebut dapat dilihat bahwa jarak benda terhadap *setpoint* tidak memiliki pengaruh yang signifikan terhadap *rise time*. Hal tersebut dikarenakan sistem menggunakan kontrol proporsional.

Pada data Gambar 4.6nampak bahwa rentang *overshoot* adalah 17 pixel sampai dengan 71 pixel. Posisi benda dapat mempengaruhi *overshoot* seiring dengan semakin jauh jarak benda terhadap *setpoint*. *Overshoot* semakin besar ketika jarak benda dengan *set point* semakin jauh.

4.4Pengujian kecepatan robot bergerak mendekati objek secara otomatis

Metode yang sama pada pengujian sebelumnya juga dapat digunakan untuk menggali data kecepatan robot. Pada sistem ini robot dirancang untuk bergerak dengan kecepatan konstan saat mendekati objek secara otomatis. Kecepatan tidak dipengaruhi jarak benda terhadap robot.

Kecepatan robot dapat dihitung dengan meletakkan robot dan benda pada jarak tertentu secara tetap. Kemudian dilakukan pencatatan waktu tempuh robot sampai dengan berhenti didepan objek. Dengan jarak yang tetap dan waktu yang diperoleh dari pengujian didapatkan data kecepatan robot. Data kecepatan robot kemudian dirata-rata dari keseluruhan percobaan. Berikut ini data kecepatan robot:

Data dari percobaan sudah sangat gamblang menyaatkan bahwa kecepatan robot rata-rata dalam mendekati objek adalah 4,38cm/s. Dengan kecepatan sekian, dalam satu jam robot hanya akan bergerak 15,7 meter.

Pada perancangan sistem ini robot dirancang bergerak secara pelan-pelan, karena kemampuan kamera dalam mengikuti objek sangat terbatas. Apabila robot bergerak terlalu cepat, setiap kali titik *tracking* didapatkan maka titik itu akan langsung hilang. Hal tersebut dikarenakan keterbatasan metode Lukas-kanade dalam *tracking* benda cepat.

BAB V

PENUTUP

5.1 Kesimpulan

Kesimpulan yang diperoleh pada tugas akhir ini adalah:

1. Pengujian tingkat keberhasilan robot mendekati secara otomatis memperlihatkan bahwa tingkat keberhasilan robot mendekati benda secara otomatis dipengaruhi jarak benda dengan robot. Jarak optimal yang didapatkan adalah 330cm dengan spesifikasi sistem yang telah dirancang dan target yang telah ditentukan.
2. Berdasarkan data yang diperoleh dari pengujian sistem, dapat disimpulkan bahwa metode *tracking* menggunakan Lukas-Kanade sangat dibataasi kecepatan. Dengan menggunakan spesifikasi sistem yang digunakan pada tugas akhir ini robot dapt melakukan *tracking* dengan kecepatan 4,33 cm/s
3. Percobaan berbagai input koordinat titik tracking yang berbeda pada sistem tidak memiliki pengaruh *rise time* dan *overshoot* yang signifikan. Sitem dapat menangani setiap input titik *tracking* dengan kecepatan *rise time* 12,2 detik sedangkan *overshoot* terbesarsistem adalah -71 pixel, yaitu setara dengan $6{,}5^{o}$. Dengan dimikian dapat dinyatakan bahwa kontrol proporsional yang digunakan pada sistem ini sudah cukup untuk memenuhi kebutuhan *tracking.*
4. Hasil pengujian beban menunjukkan bahwa dengan menggunakan spesifikasi sistem yang digunakan robot dapat mengangkut beban hingga 217 gram

5.2 Saran

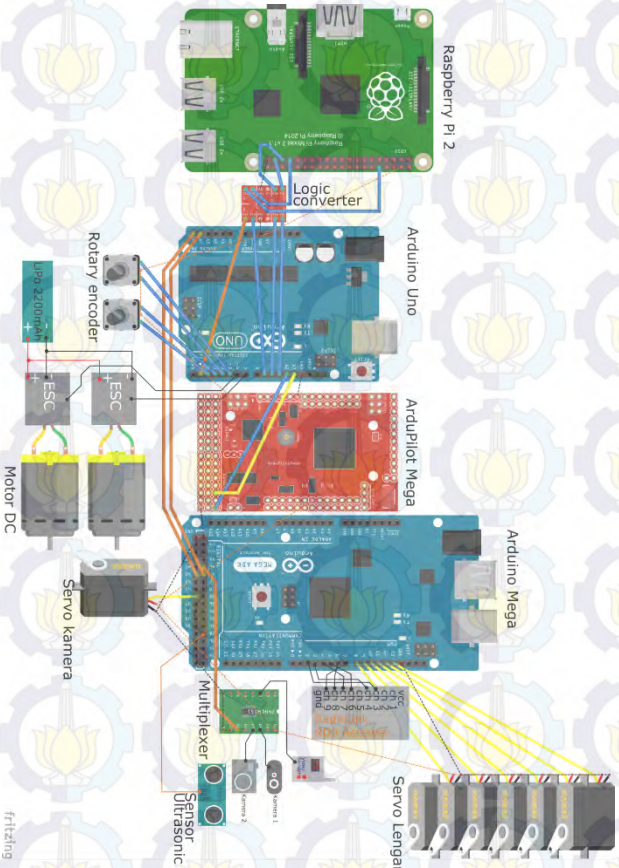
Setelah melakukan pengujian sistem, terdapat beberapa saran untuk pengembangan sistem supaya bisa menjadi lebih baik:

1. Sistem dapat bekerja lebih optimal menggunakan unit proses yang memiliki kecepatan proses lebih baik.
2. Sistem juga dapat bekerja lebih optimal menggunakan kamera dengan kecepatan *frame* lebih tinggi dan sudut pandang yang lebih flexible seperti kamera *PTZ.*

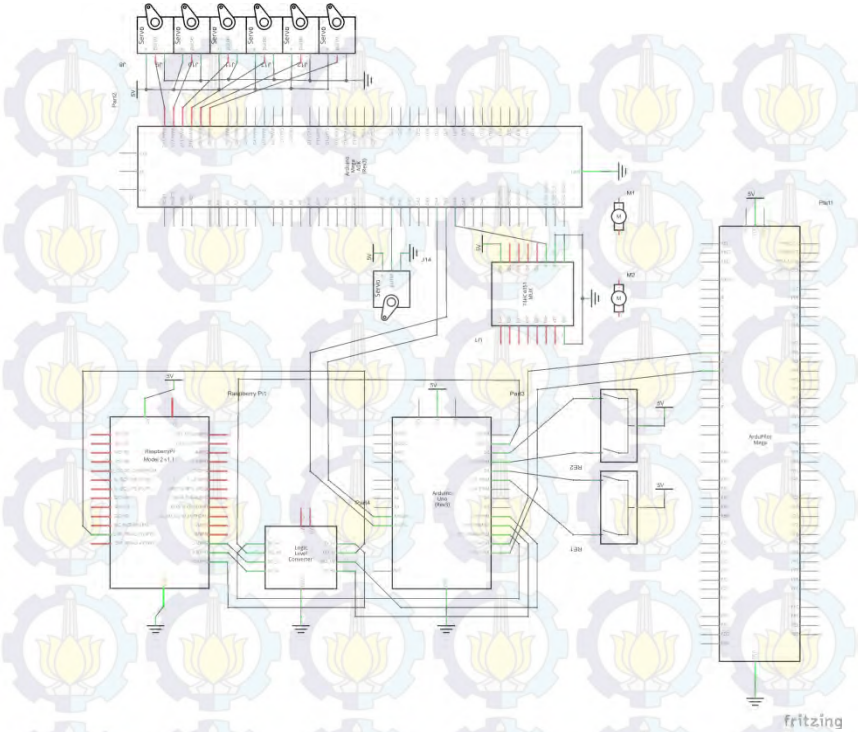
3. Sistem juga dapat bekerja lebih optimal menggunakan lengan robot yang dirangkai menggunakan motor servo dengan daya yang lebih besar. Lengan robot juga dapat dikembangkan sehingga lengan robot dapat mengambil benda secara otomatis.
4. Robot pada tugas akhir ini dapat dikembangkan menggunakan roda yang lebih besar sehingga dapat meningkatkan kinerja robot diluar ruangan.

LAMPIRAN

Ilustrasi skematik sistem:



Skematik sistem:



Berikut ini program pada Arduino Mega:

```
#include <Servo.h>
#define thromin 1373//analog kanan bawah
#define thromax 2497//analog kanan atas

#define MODE_BIASA     1
#define MODE_RASPBERRY  2
#define MODE_TRACKING   3
int mode = MODE_BIASA;

//<-------variable sensor Ping
unsigned long echo = 0;
int ultraSoundSignal = 38; // Ultrasound signal pin
unsigned long ultrasoundValue = 0;
int nilaiPING=0;
boolean ping_ok = 1;
boolean lurus_1x = 1;

//<------variable motor servo
Servo ser1,ser2,ser3,ser4,ser5,ser6,serCam; //variabel 8 servo; 6 servo
lengan 2 servo Tank

//<------variable receiver
int input1,input2,input3,input4,
input5,input6,input7,input8,input9; //input dari remote
int serIn[9];

int out1,out2,out3,out4,
out5,out6,out7,out8,out9; //output menuju servo
double serOut[9]={5,90,3,90,90}; //arm initial position

int res1,res2,res3,res4,
res5,res6,res7,res8,res9; //buffer untuk mengurangi
ketelitian/resolution
int res[9]={0};

//<-------panjang lengan dalam cm
```

```
float a2=  10.3;
float a3=  9.7;
float a4=  15.7;
float d=   4.3;
//============Theta 3============//input dalam radian,hasil
dalam radian
double theta3 (double x0, double y0, double z0, double phi0){ //input
dalam radian
double r =(-1*(acos(
(
((sqrt((x0*x0)+(y0*y0)))- a4*cos(phi0))*((sqrt((x0*x0)+(y0*y0)))-
a4*cos(phi0))+
((z0-d)-a4*sin(phi0))*((z0-d)-a4*sin(phi0))-
(a2*a2)-(a3*a3)
)/(2*a2*a3)
)));
if(isfinite(r))return constrain(r,-2.44, 0);// pembatasan -140 s.d 0
}

//=========Theta 2==========//input dalam radian,hasil dalam
radian
double theta2 (double x0, double y0, double z0, double phi0){
double r =(atan2 (
((z0-d)-a4*sin(phi0)),((sqrt((x0*x0)+(y0*y0)))- a4*cos(phi0))
)-
        atan2 (
(a3*sin(theta3(x0,y0,z0,phi0))),(a2+(a3*cos(theta3(x0,y0,z0,phi0))))
));
if(isfinite(r))return constrain(r,0, 1.57); // pembatasan 0 s.d 90
}

//==========Theta 4========//input dalam radian,hasil dalam
radian
double theta4 (double x0, double y0, double z0, double phi0){
double r =(phi0
-theta2(x0,y0,z0,phi0)
-theta3(x0,y0,z0,phi0));
if(isfinite(r))return constrain(r,-2.44, 0); //pembatasan -140 s.d 0
}
```

```
//========Theta 1=======//input dalam radian,hasil dalam radian
double theta1(double x0, double y0){
double r =(atan2 (y0,x0));
if(isfinite(r))return constrain(r,-1.57, 1.57); //pembatasan -180 s.d 180
}

void execute(double x0, double y0, double z0, double phi0){
    phi0=((phi0)*PI/180);
ser1.write(170-(180/PI)*theta2(x0,y0,z0,phi0));
ser2.write((180/PI)*theta2(x0,y0,z0,phi0));//tambah 180, yg bawah
juga

    ser3.write(-(180/PI)*theta3(x0,y0,z0,phi0));//tambah 180
ser4.write(-(180/PI)*theta4(x0,y0,z0,phi0));
}
//==========PING SENSOR
unsigned long ping()
{
pinMode(ultraSoundSignal, OUTPUT); // Switch signalpin to output
digitalWrite(ultraSoundSignal, LOW); // Send low pulse
delayMicroseconds(2); // Wait for 2 microseconds
digitalWrite(ultraSoundSignal, HIGH); // Send high pulse
delayMicroseconds(5); // Wait for 5 microseconds
digitalWrite(ultraSoundSignal, LOW); // Holdoff
pinMode(ultraSoundSignal, INPUT); // Switch signalpin to input
digitalWrite(ultraSoundSignal, HIGH); // Turn on pullup resistor
echo= pulseIn(ultraSoundSignal, HIGH); //Listen for echo
ultrasoundValue=(echo / 58.138)*.39*2; //convert to CM then to
inches
return ultrasoundValue;
}
//==========analog input average
double avg(double input, int count){
double output=0;
double avgbuff=0; //buffer untuk menyimpan data rata-rata
for(int i=0; i<count; i++){
avgbuff= input;
```

```
output=output+avgbuff;
}
return output/(count);
}
//==========SETUP
void setup(){
ser1.attach(8);//servo base // theta 2 reverse
ser2.attach(9);//servo base // theta 2
ser3.attach(10);// theta 3
ser4.attach(11);// theta 4
ser5.attach(12);// griper twist
ser6.attach(13);// capit
serCam.attach(39);//servo kamera
  //-------------------------pin switch kamera
pinMode(46,OUTPUT);
digitalWrite(46,HIGH);
  //-------------------------mode input pin 2-7
for(int i=2; i<=7; i++){
pinMode(i,INPUT);
digitalWrite(i,HIGH);
}
  //-------initialisasi pin ultasonic, pin 38
pinMode(ultraSoundSignal,OUTPUT);
  //----------initial arm position
execute(serOut[0],0,serOut[2],-90);
  //----pin Mode selection dan Ping Sensor
pinMode(43, OUTPUT);      //UNO pin 8
digitalWrite(43, LOW);
pinMode(44, OUTPUT);      //UNO pin A4
digitalWrite(44, LOW);
pinMode(45, OUTPUT);      //UNO pin A5
digitalWrite(45, LOW);
  //Serial.begin(9600);
}
//=====LOOP
void loop (){
    //--------Switch MODE
serIn[3]=avg(pulseIn (5, HIGH, 35000),3); //Pin 5, Receiver 6, switch
TRACKING MODE
```

```
    //Serial.println(serIn[3]);
if((serIn[3]< thromin +150 )&&(serIn[3]>0)){  //saklar ke atas
digitalWrite(46,HIGH);   //nyalakan kamera biasa
mode= MODE_BIASA;       //mode biasa
digitalWrite(44,LOW);
digitalWrite(45,HIGH);
}
elseif((serIn[3]> thromin +250 )&&(serIn[3]< thromax -250 )){
digitalWrite(46,LOW);  //nyalakan kamera raspberry
mode= MODE_RASPBERRY;  //ganti mode raspberry
digitalWrite(44,HIGH);
digitalWrite(45,LOW);
}
elseif(serIn[3]> thromax -150 ){
digitalWrite(46,LOW);  //nyalakan kamera raspberry
mode= MODE_TRACKING;
digitalWrite(44,HIGH);
digitalWrite(45,HIGH);
}
else{
digitalWrite(46,HIGH);  //nyalakan kamera raspberry
}

    //---------Manual navigation
if(mode==MODE_BIASA){
      //---------PING, berhenti ketika jarak sudah
nilaiPING= ping();
if(nilaiPING < 40)digitalWrite(43,LOW);
else digitalWrite(43,HIGH);

serIn[0]=avg(pulseIn (2, HIGH, 35000),3); //Pin 2, Receiver 3, Gerak
X //===========channel<<==1==2==5==>>dudah di booking
APM
serIn[1]=avg(pulseIn (3, HIGH, 35000),3); //Pin 3, Receiver 4, Servo
kamera
serIn[2]=avg(pulseIn (4, HIGH, 35000),3); //Pin 4, Receiver 7, Gerak
Z
serIn[4]=avg(pulseIn (6, HIGH, 35000),3); //Pin 6, Receiver 8, Griper
twist
```

```
serIn[5]=avg(pulseIn (7, HIGH, 35000),3); //Pin 7, Receiver 9, Griper capit
        //--------Gerak X
if(serIn[0]> thromax-300){
serOut[0]=serOut[0]-0.5;
serOut[0]=constrain(serOut[0],3, 22);
execute(serOut[0],0,serOut[2],-90);}
elseif((serIn[0]< thromin+300)&&(serIn[0]>0)){
serOut[0]=serOut[0]+0.5;
serOut[0]=constrain(serOut[0],3, 22);
execute(serOut[0],0,serOut[2],-90);
}
else execute(serOut[0],0,serOut[2],-90);

      //---------Gerak Z
if(serIn[2]> thromax-300){
serOut[2]=serOut[2]-1;
serOut[2]=constrain(serOut[2],-20, 5);
execute(serOut[0],0,serOut[2],-90);
}
elseif((serIn[2]< thromin+300)&&(serIn[2]>0)){
serOut[2]=serOut[2]+0.5;
serOut[2]=constrain(serOut[2],-20, 5);
execute(serOut[0],0,serOut[2],-90);
if(serOut[2]<-7)digitalWrite(43,LOW);
else digitalWrite(43,HIGH);
}
else execute(serOut[0],0,serOut[2],-90);

      //---------gripper twist
if(serIn[4]> thromax-400){
serOut[4]=serOut[4]+5;
serOut[4]=constrain(serOut[4],0, 180);
ser5.write(serOut[4]);
}
elseif((serIn[4]< thromin+400)&&(serIn[4]>0)){
serOut[4]=serOut[4]-5;
serOut[4]=constrain(serOut[4],0, 180);
ser5.write(serOut[4]);
```

```
}

      //---------Capit
if(serIn[5]> thromax -150 ) ser6.write(180);
elseif((serIn[5]< thromin +150 )&&(serIn[5]>0)) ser6.write(110);
else ser6.write(110);

     //--------gerakkan servo kamera
if((serIn[1]>0)&&(serIn[1]<thromin+200)){
serOut[1]=serOut[1]+5;
serOut[1]=constrain(serOut[1],0, 180);
serCam.write(serOut[1]);
}
elseif(serIn[1]>thromax-200){
serOut[1]=serOut[1]-5;
serOut[1]=constrain(serOut[1],0, 180);
serCam.write(serOut[1]);
}
}
if(mode==MODE_TRACKING){
      //-PING, berhenti ketika jarak sudah
nilaiPING= ping();
if(nilaiPING < 7)digitalWrite(43,LOW);
else digitalWrite(43,HIGH);
}
}
```

Berikut ini program pada Arduino Uno:

```
#include <Servo.h>
#define APM_MAX_KIRI 2511
#define APM_MIN_KIRI 1354
#define APM_MAX_KANAN 2501
#define APM_MIN_KANAN 1372

#define encoderKiriPinA  2
#define encoderKananPinA  3
#define MODE_BIASA      1
#define MODE_RASPBERRY  2
```

```
#define MODE_TRACKING   3
int mode = MODE_BIASA;

volatile unsigned int encoderKiriPos = 0;
volatile unsigned int encoderKananPos = 0;
long newpositionKiri,newpositionKanan;
long oldpositionKiri = 0, oldpositionKanan=0;
unsigned long newtimeKiri, newtimeKanan;
unsigned long oldtimeKiri = 0,oldtimeKanan = 0;
long velKiri, velKanan,velSetKanan=500,velSetKiri=500;
long maxSpeed=1000;
float potKanan=0, potKiri=0;

Servo motorKiri,motorKanan;
int apmKiri, apmKanan;
boolean ping_ok, interrupt_ok;
unsigned int bacaRaspberry;

void setup(){
pinMode(encoderKiriPinA, INPUT);
digitalWrite(encoderKiriPinA, HIGH);  pinMode(encoderKananPinA,
INPUT);
digitalWrite(encoderKananPinA, HIGH);
pinMode(4, INPUT);     //kiri
digitalWrite(4, HIGH);
pinMode(5, INPUT);     //kanan
digitalWrite(5, HIGH);
  //-------------------------servo
motorKiri.attach(6);// servo steering
motorKanan.attach(7);// servo accelerating
  //-------------------------pin Click, Clear, Kanan, Kiri, Atas, Bawah
pinMode(11,OUTPUT);
digitalWrite(11,HIGH);
pinMode(10,OUTPUT);
digitalWrite(10,LOW);
pinMode(9,OUTPUT);
digitalWrite(9,LOW);
  //-------------------------pin sensor PING
pinMode(8, INPUT);
```

```
digitalWrite(8, HIGH);
  //------------------------pin input APM
pinMode(12,INPUT);        //APM 1, motor Kiri
digitalWrite(12,HIGH);
pinMode(13,INPUT);        //APM 3, motor Kanan
digitalWrite(13,HIGH);
  //------------------------pin input MODE SELECT
pinMode(A4, INPUT);
digitalWrite(A4, HIGH);
pinMode(A5, INPUT);
digitalWrite(A5, HIGH);
  Serial.begin (9600);
}

void loop (){
  //delay(500);
if(digitalRead(8)==LOW){
    ping_ok=0;
}
elseif(digitalRead(8)==HIGH)ping_ok=1;
if((digitalRead(A4)==LOW)&&(digitalRead(A5)==HIGH))mode=M
ODE_BIASA;
elseif((digitalRead(A4)==HIGH)&&(digitalRead(A5)==LOW))mode
=MODE_RASPBERRY;
elseif((digitalRead(A4)==HIGH)&&(digitalRead(A5)==HIGH))mode
=MODE_TRACKING;

if(mode==MODE_BIASA){
digitalWrite(11,HIGH);  //clear
digitalWrite(10,HIGH);
digitalWrite(9,LOW);
interruptMati();
apmKiri=avg(pulseIn (12, HIGH, 35000),3); //Pin 12, APM 1, motor
Kiri
apmKanan=avg(pulseIn (13, HIGH, 35000),3); //Pin 13, APM 3,
motor Kanan
apmKiri=map(apmKiri, APM_MIN_KIRI, APM_MAX_KIRI,
550,2400);
```

```
apmKanan=map(apmKanan, APM_MIN_KANAN,
APM_MAX_KANAN, 550,2400);
if(!ping_ok){
apmKiri=  constrain(apmKiri,550,1400);
apmKanan=  constrain(apmKanan,550,1400);
}
motorKiri.writeMicroseconds(apmKiri);//balik
motorKanan.writeMicroseconds(apmKanan);//balik
}
if(mode==MODE_TRACKING){
digitalWrite(11,HIGH);  //click
digitalWrite(10,LOW);
digitalWrite(9,HIGH);
if((Serial.available()>0)&&(ping_ok)){
bacaRaspberry=Serial.read();
}
interruptAktif();
kontrolTracking();
kecepatanKanan();
kecepatanKiri();
if(!ping_ok){
motorKanan.writeMicroseconds(1460);
motorKiri.writeMicroseconds(1460);
}
else{
motorKanan.writeMicroseconds(1500+potKanan);
motorKiri.writeMicroseconds(1500+potKiri);
}
delay(10);
}
if(mode == MODE_RASPBERRY){
interruptMati();
apmKiri=avg(pulseIn (12, HIGH, 35000),3); //Pin 12, APM 1, motor
Kiri
apmKanan=avg(pulseIn (13, HIGH, 35000),3); //Pin 13, APM 3,
motor Kanan
motorKanan.writeMicroseconds(1460);//diam
motorKiri.writeMicroseconds(1460);//diam
potKiri=0; potKanan=0;
```

```
if((apmKiri < APM_MIN_KIRI +150)
&&(apmKanan < APM_MIN_KANAN +150)
&&(mode==MODE_RASPBERRY)){ //tuas mode X ke KIRI
raspberry
digitalWrite(11,LOW);  //kiri
digitalWrite(10,HIGH);
digitalWrite(9,LOW);
}
elseif((apmKiri > APM_MAX_KIRI -500)
&&(apmKanan > 1684 -100 )
&&(apmKanan < 1684 +100 )
&&(mode==MODE_RASPBERRY)){ //tuas mode X ke KANAN
raspberry
digitalWrite(11,LOW);  //kanan
digitalWrite(10,LOW);
digitalWrite(9,HIGH);
}
elseif((apmKanan > APM_MAX_KANAN -500)
&&(apmKiri > 1716 -100 )
&&(apmKiri < 1716 +100 )
&&(mode==MODE_RASPBERRY)){ //tuas mode Y ke ATAS
raspberry
digitalWrite(11,LOW);  //atas
digitalWrite(10,HIGH);
digitalWrite(9,HIGH);
}
elseif((apmKiri > APM_MAX_KIRI -150)
&&(apmKanan > APM_MAX_KANAN -150)
&&(mode==MODE_RASPBERRY)){ //tuas mode Y ke BAWAH
raspberry
digitalWrite(11,LOW);  //bawah
digitalWrite(10,LOW);
digitalWrite(9,LOW);
}
else{
digitalWrite(11,HIGH);  //diam
digitalWrite(10,LOW);
digitalWrite(9,LOW);
}
```

```
}
}

void doEncoderKiri(){
if((digitalRead(encoderKiriPinA)== HIGH)&&(digitalRead(4)==
HIGH)){
encoderKiriPos++;
}else{
encoderKiriPos--;
}
}
void doEncoderKanan(){
if((digitalRead(encoderKananPinA)== HIGH)&&(digitalRead(5)==
HIGH)){
encoderKananPos++;
}else{
encoderKananPos--;
}
}
void kecepatanKiri (){
newpositionKiri= encoderKiriPos;
newtimeKiri= micros();
velKiri=((oldpositionKiri-newpositionKiri)*1000000 /(newtimeKiri-
oldtimeKiri));
oldpositionKiri= newpositionKiri;
oldtimeKiri= newtimeKiri;
if(velKiri< velSetKiri){
potKiri=potKiri+(0.1/*(velSetKiri-velKiri)*(100/maxSpeed)*/);
//kontrol Proporsional kecepatan
}
elseif(velKiri> velSetKiri){
potKiri=potKiri-(0.1/*(velKiri-velSetKiri)*(100/maxSpeed)*/);
//kontrol Proporsional kecepatan
}
}
void kecepatanKanan (){
newpositionKanan= encoderKananPos;
newtimeKanan= micros();
```

```
velKanan=((newpositionKanan-oldpositionKanan)*1000000
/(newtimeKanan-oldtimeKanan));
oldpositionKanan= newpositionKanan;
oldtimeKanan= newtimeKanan;
if(velKanan< velSetKanan){
potKanan=potKanan+(0.1/*(velSetKanan-
velKanan)*(100/maxSpeed)*/);  //kontrol Proporsional kecepatan
}
elseif(velKanan> velSetKanan){
potKanan=potKanan-(0.1/*(velKanan-
velSetKanan)*(100/maxSpeed)*/);  //kontrol Proporsional kecepatan
}
}
void kontrolTracking(){
if(bacaRaspberry<=33){velSetKanan=0;velSetKiri=0;}
elseif((bacaRaspberry<145)&&(bacaRaspberry>33)){//belok kiri
   //bacaRaspberry=145-bacaRaspberry;
velSetKiri=(bacaRaspberry-34)*(maxSpeed/111);      //kontrol
Proporsional kecepatan
velSetKanan=maxSpeed;
}
elseif(bacaRaspberry>145){//belok kanan
   //bacaRaspberry=255-bacaRaspberry;
velSetKanan=(255-bacaRaspberry)*(maxSpeed/111);       //kontrol
Proporsional kecepatan
velSetKiri=maxSpeed;
}
}
//=analog input average==
double avg(double input, int count){
double output=0;
double avgbuff=0; //buffer untuk menyimpan data rata-rata
for(int i=0; i<count; i++){
avgbuff= input;
output=output+avgbuff;
}
return output/(count);
}
//=Prosedur mengaktifkan interrupt
```

```
void interruptAktif(){
if(interrupt_ok==0){
attachInterrupt(0, doEncoderKiri, RISING);  // encoder pin on
interrupt 0 - pin 20
attachInterrupt(1, doEncoderKanan, RISING);
        interrupt_ok=1;
}
}
void interruptMati(){
if(interrupt_ok==1){
detachInterrupt(0);
detachInterrupt(1);
        interrupt_ok=0;
}
}
```

Berikut ini program pada Raspberry Pi:

```
/* Demo of modified Lucas-Kanade optical flow algorithm.
   See the printf below */

#ifdef _CH_
#pragma package <opencv>
#endif

#define CV_NO_BACKWARD_COMPATIBILITY

#ifndef _EiC
#include "cv.h"
#include "highgui.h"
#include <stdio.h>
#include <ctype.h>
#include <wiringPi.h>
#include <wiringSerial.h>
#endif

IplImage *image = 0,*grey = 0,*prev_grey = 0,*pyramid =
0,*prev_pyramid = 0,*swap_temp;

int win_size = 10;
```

```
const int MAX_COUNT = 500;//?
CvPoint2D32f*points[2]={0,0},*swap_points;
char* status = 0;
int count = 0;
int need_to_init = 0;
int night_mode = 0;
int flags = 0;
int add_remove_pt = 0;
CvPoint pt;
CvPoint ptL=cvPoint(100,100);


void on_mouse( int event, int x, int y, int flags, void* param )
{
if(!image )
return;

if( image->origin )
      y = image->height - y;

if( event == CV_EVENT_LBUTTONDOWN )
{
pt= cvPoint(x,y);
      add_remove_pt = 1;
}
}


int main( int argc, char** argv )
{
wiringPiSetup();
int fd = serialOpen("/dev/ttyACM0",9600);
char getC;

   CvCapture* capture = 0;

if( argc == 1 ||(argc == 2 && strlen(argv[1])== 1 &&
isdigit(argv[1][0])))
capture= cvCaptureFromCAM( argc == 2 ? argv[1][0]- '0' : 0 );
```

```
elseif( argc == 2 )
capture= cvCaptureFromAVI( argv[1]);

if(!capture )
{
fprintf(stderr,"Could not initialize capturing...\n");
return-1;
}

    /* print a welcome message, and the OpenCV version */
printf("Welcome to lkdemo, using OpenCV version %s
(%d.%d.%d)\n",
        CV_VERSION,
        CV_MAJOR_VERSION, CV_MINOR_VERSION,
CV_SUBMINOR_VERSION);

printf( "Hot keys: \n"
          "\tESC - quit the program\n"
          "\tr - auto-initialize tracking\n"
          "\tc - delete all the points\n"
          "\tn - switch the \"night\" mode on/off\n"
          "To add/remove a feature point click it\n" );

cvNamedWindow( "LkDemo", 0 );
cvSetMouseCallback( "LkDemo", on_mouse, 0 );

for(;;)
{
      IplImage* frame = 0;
int i, k, c;

frame= cvQueryFrame( capture );
if(!frame )
break;

if(!image )
{
          /* allocate all the buffers */
image= cvCreateImage( cvGetSize(frame), 8, 3 );
```

```
image->origin = frame->origin;
grey= cvCreateImage( cvGetSize(frame), 8, 1 );
        prev_grey =cvCreateImage( cvGetSize(frame), 8, 1 );
pyramid= cvCreateImage( cvGetSize(frame), 8, 1 );
        prev_pyramid =cvCreateImage( cvGetSize(frame), 8, 1 );
points[0]=(CvPoint2D32f*)cvAlloc(MAX_COUNT*sizeof(points[0][
0]));//?
points[1]=(CvPoint2D32f*)cvAlloc(MAX_COUNT*sizeof(points[0][
0]));//?
status=(char*)cvAlloc(MAX_COUNT);//?
flags= 0;//?
ptL= cvPoint((frame->width)/2,(frame->height)/2);
}

cvCopy( frame, image, 0 );
cvCvtColor( image, grey, CV_BGR2GRAY );

if( night_mode )
cvZero( image );

if( need_to_init )
{
        /* automatic initialization */
        IplImage* eig =cvCreateImage( cvGetSize(grey), 32, 1 );
        IplImage* temp =cvCreateImage( cvGetSize(grey), 32, 1 );
double quality = 0.01;
double min_distance = 10;

count= MAX_COUNT;
cvGoodFeaturesToTrack( grey, eig, temp, points[1],&count,
quality, min_distance, 0, 3, 0, 0.04 );
cvFindCornerSubPix( grey, points[1], count,
cvSize(win_size,win_size), cvSize(-1,-1),
cvTermCriteria(CV_TERMCRIT_ITER|CV_TERMCRIT_EPS,20,0.0
3));
cvReleaseImage(&eig );
cvReleaseImage(&temp );

        add_remove_pt = 0;
```

```
}
elseif( count > 0 )
{
cvCalcOpticalFlowPyrLK( prev_grey, grey, prev_pyramid, pyramid,
points[0], points[1], count, cvSize(win_size,win_size), 3, status, 0,
cvTermCriteria(CV_TERMCRIT_ITER|CV_TERMCRIT_EPS,20,0.0
3), flags );
flags|= CV_LKFLOW_PYR_A_READY;
for( i = k = 0; i < count; i++)
{
if( add_remove_pt )
{
double dx = pt.x - points[1][i].x;
double dy = pt.y - points[1][i].y;

if( dx*dx + dy*dy <= 25 )
{
                add_remove_pt = 0;
continue;
}
}

if(!status[i])
continue;

points[1][k++]= points[1][i];
            cvCircle( image, cvPointFrom32f(points[1][i]), 3,
CV_RGB(0,255,0),-1, 8,0);
}
count= k;
}

cvCircle(image, ptL, 8, CV_RGB(255,0,0), 3, 8,0);
cvRectangle(image,cvPoint(280,200),
cvPoint(360,280),CV_RGB(255,255,255),1,8,0);
if((points[1][1].x>0)&&(points[1][1].x<200)) printf("z\n");
elseif(points[1][1].x>280) printf("x\n");
if( add_remove_pt && count < MAX_COUNT )
{
```

```
points[1][count++]= cvPointTo32f(pt);
cvFindCornerSubPix( grey, points[1]+ count - 1, 1,
cvSize(win_size,win_size), cvSize(-1,-1),
cvTermCriteria(CV_TERMCRIT_ITER|CV_TERMCRIT_EPS,20,0.0
3));
        add_remove_pt = 0;
}

      CV_SWAP( prev_grey, grey, swap_temp );
      CV_SWAP( prev_pyramid, pyramid, swap_temp );
      CV_SWAP( points[0], points[1], swap_points );
      need_to_init = 0;
cvShowImage( "LkDemo", image );
if(serialDataAvail(fd)>0) getC=serialGetchar(fd);
switch((char) getC )
{
case 'a':
        ptL.x=ptL.x-9;
printf("%c\n",getC);
break;
case 'w':
        ptL.y=ptL.y-9;
break;
case 's':
        ptL.y=ptL.y+9;
break;
case 'd':
        ptL.x=ptL.x+9;
break;
case 'q':
pt= ptL;
        add_remove_pt = 1;
break;
case 'e':
ptL=cvPoint((image->width)/2,(image->height)/2);
break;
default:
break;
}
```

```
        c =cvWaitKey(10);
    if((char)c == 27 )
    break;
    switch((char) c )
    {
    case 'r':
            need_to_init = 1;
    break;
    case 'c':
    count= 0;
    break;
    case 'n':
            night_mode ^= 1;
    break;
    default:
    break;
    }
    }

    cvReleaseCapture(&capture );
    cvDestroyWindow("LkDemo");

    return 0;
    }
    #ifdef _EiC
    main(1,"lkdemo.c");
    #endif
```

DAFTAR PUSTAKA

[1]Departemen Pendidikan Nasional, "KBBI,"badanbahasa.kemdikbud.go.id, 2008

[2] John J. Craig, "Introduction to Robotics: Mechanics and Control",3rd Edition, Prentice-Hall. 2004,

[3] Tele Radio AB. "What is industrial remote control,"www.tele-radio.com,2014

[4] “PWM Controller”, <URL: http://www.best-microcontroller-projects.com/pwm-pic.html>

[5] Fujiwara, Y."Self-Synchronizing Pulse Position Modulation With Error Tolerance," IEEE Explore.2013.

[6] Patrick Bertagna, "How does a GPS tracking system work?”,<URL: www.eetimes.com>, 2010.

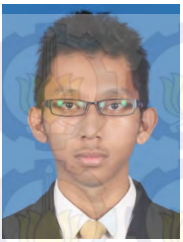
[7]u-blox Holding AG ”NEO-6 - Data Sheet”, 2011

[8] Gary Bradski “Learning OpenCV”, O’Reilly Media,2008

[9] Dani Prasetyawan, “Rancang Bangun Sistem PelacakanOtomatis Dan Penguncian Sasaran PadaPertahanan Statis Berbasis PengolahCitra”, 2015.

Halaman Ini Sengaja Dikosongkan

RIWAYAT PENULIS



Muhammad Qomaruzzaman lahir pada tanggal 22 Oktober 1992. Penulis lahir dan dibesarkan di Gresik. Penulis menempuh sekolah dasar di MI Ma'arif Assa'adah, kemudian penulis melanjutkan pendidikan di MTs Ma'arif Assa'adah 1. Sembilan tahun penulis menempuh pendidikan di Gresik, penulis kemudian melanjutkan pendidikan di SMAN 3 Malang. Setelah menamatkan pendidikan di Malang, penulis melanjutkan pendidikan di Surabaya, yaitu Institut Teknologi Sepulih Nopember. Tanpa pengetahuan yang cukup tentang apa yang akan dihadapai didunia perkuliahan, penulis mengasah pengetahuan tentang elektronika baik dibangku kuliah maupun melalui komunitas mahasiswa. Selama kuliah, penulis pun aktif membantu kegiatan laboratorium seperti praktikum dan pelatihan.

Email: mqomaruzzaman@gmail.com