UNIVERSITAS INDONESIA

RANCANG BANGUN COMPLIANCE CONTROL PADA TANGAN ROBOT PENERIMA TAMU

SKRIPSI

Diajukan sebagai salah satu syarat memperoleh gelar sarjana

Disusun Oleh :

HARIS KASMINTO AJI

0806330945

DEPARTEMEN TEKNIK ELEKTRO

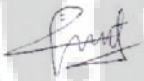
FAKULTAS TEKNIK

UNIVERSITAS INDONESIA

JUNI 2012

HALAMAN PERNYATAAN ORISINALITAS

Skripsi ini adalah hasil karya saya sendiri,
dan semua sumber baik yang dikutip maupun dirujuk
telah saya nyatakan dengan benar.

Nama	**: Haris Kasminto Aji**
NPM	**: 0806330945**
Tanda Tangan	**:** 
Tanggal	**: 25 Juni 2012**

HALAMAN PENGESAHAN

Skripsi ini diajukan oleh :

Nama :Haris Kasminto Aji
NPM : 0806330945
Program Studi :Teknik Elektro
Judul Skripsi : Rancang Bangun Compliance Control Pada Tangan Robot Penerima Tamu

Telah berhasil dipertahankan di hadapan Dewan Penguji dan diterima sebagai bagian persyaratan yang diperlukan untuk memperoleh gelar Sarjana Teknik pada Program Studi Teknik Elektro, Fakultas Teknik, Universitas Indonesia.

DEWAN PENGUJI

Pembimbing :

Dr. Abdul Muis, ST., M.Eng ( )

Penguji 1 :

Ir.Wahidin Wahab M.Sc,Ph.D ( )

Penguji 2 :

Prof. Dr.Eng. Drs. Benyamin Kusumoputro M.Eng. ( )

Ditetapkan di : Depok

Tanggal : 25 Juni 2012

KATA PENGANTAR

Segala puji syukur kehadirat Tuhan Yang Maha Esa, karena atas nikmat, berkat, dan rahmat-Nya penulisan skripsi ini dapat terselesaikan. Penulisan skripsi ini dilakukan dalam rangka memenuhi salah satu syarat mencapai gelar sarjana Teknik Elektro pada Fakultas Teknik Universitas Indonesia. Saya menyadari bahwa tanpa bantuan dan bimbingan dari berbagai pihak, sangatlah sulit bagi saya untuk menyelesaikan skripsi ini. Oleh karena itu, saya mengucapkan terima kasih kepada:

1. Dr. Abdul Muis S.T., M.Eng, selaku dosen pembimbing, serta dosen-dosen lainnya, yang telah menyediakan waktu, tenaga, dan pikiran untuk mengarahkan saya dalam penyusunan laporan seminar ini;
2. Orang tua dan keluarga saya yang telah memberikan kasih sayang dan semangat serta bantuan berupa dukungan material, moral, dan spiritual;
3. Teman - teman mahasiswa Teknik Elektro program Reguler khususnya angkatan 2008 yang sama - sama saling memberi dorongan dan semangat dalam mengerjakan penulisan skripsi ini.
4. Kepada rekan - rekan lain yang tidak dapat disebutkan satu persatu yang turut membantu dan memberikan dukungan kepada penulis.

Semoga Tuhan Yang Maha Esa membalas semua kebaikan semua pihak yang membantu dan semoga skripsi ini dapat bermanfaat untuk pengembangan ilmu.

HALAMAN PERNYATAAN PERSETUJUAN PUBLIKASI TUGAS AKHIR UNTUK KEPENTINGAN AKADEMIS

Sebagai civitas akademik Universitas Indonesia, saya yang bertanda tangan di bawah ini:

Nama	: Haris Kasminto Aji
NPM	: 0806330945
Program Studi	: Teknik Elektro
Departemen	: Teknik Elektro
Fakultas	: Teknik
Jenis Karya	: Skripsi

Demi pengembangan ilmu pengetahuan, menyetujui untuk memberikan kepada Universitas Indonesia **Hak Bebas Royalti Noneksklusif (*Non-exclusive Royalty Free Right*)** atas karya ilmiah saya yang berjudul:

RANCANG BANGUN COMPLIANCE CONTROL PADA TANGAN ROBOT PENERIMA TAMU

beserta perangkat yang ada. Dengan Hak Bebas Royalti Noneksklusif ini Universitas Indonesia berhak menyimpan, mengalihmedia/formatkan, mengelola dalam bentuk pangkalan data (*database*), merawat, dan mempublikasikan tugas akhir saya selama tetap mencantumkan nama saya sebagai penulis/pencipta dan sebagai pemilik Hak Cipta.

Demikian pernyataan ini saya buat dengan sebenarnya.

Dibuat di : Depok

Pada tanggal : 13 Juni 2012

Yang menyatakan

(Haris Kasminto Aji)

ABSTRAK

Nama : Haris Kasminto Aji
Program Studi : Teknik Elektro
Judul :Rancang Bangun Compliance Control Pada Tangan Robot Penerima Tamu

Akhir-akhir ini kemajuan teknologi robot sangat pesat. Salah satunya adalah robot yang mempunyai kemampuan berinteraksi dengan manusia dan lingkungan secara langsung, tidak membahayakan, dan lebih bersahabat. Pada penelitian ini, akan diimplemetasikan *compliance control* pada tangan robot dengan mendapatkan informasi *force* yang diatur dari *compliance strategy* yang memberikan robot sebuah *real intelligent* sehingga robot dapat *compliance* terhadap lingkungan. Pada penelitian ini dimanfaatkan force/torsi (F/T) feedback dari dynamixel rx-24, dan current feedback dari motor DC. Dengan menerapkan *Resolved Motion Rate Control (RMRC)*, *trajectory planning* dapat dibuat dan setiap waktu, posisi, kecepatan, serta gaya aktual pada setiap joint dapat terekam melalui sensor yang ada pada dynamixel. Untuk memperoleh dinamika sistem digunakan *Newton-Euler equation*. Hasilnya, Implementasi compliance control pada tangan robot telah berhasil walaupun masih kaku dan kurang presisi.

Kata kunci : compliance control, dynamixel rx-24, Resolved Motion Rate Control (RMRC), Newton Euler Equation.

ABSTRACK

Name : Haris Kasminto Aji

Study Program : Electrical Engineering

Title :Development Through Compliance Control at Receptionist Robot Hand

Nowdays, technology of robotic increase rapidly. One of them is robot which have ability to interact with human and environment directly, not dangerous, and more friendly. In this research, designed robot hand using dynamixel as actuator robot arm and DC gearmotor as actuator hand fingers. In this research, implemented *compliance control* with get force information from *compliance strategy* which give robot real intelligent to solve the task well. In this research is used torque feedback from dynamixel rx-24 and current feedback from DC gearmotor. With implement Resolved Motion Rate Control (RMRC), robot hand can move follow instruction given and every time, position, velocity, and actual force every joint can be recorded by dynamixel's sensors. With using *Newton-Euler equation* about dynamic, so can be gotten force equation in every moving. The result, implementation of compliance control in hand robot is successful although still rough and bad presision.

Key word : compliance control, dynamixel rx-24, Resolved Motion Rate Control (RMRC), Newton Euler Equation.

DAFTAR ISI

HALAMAN PERNYATAAN ORISINALITAS…… ii
HALAMAN PENGESAHAN…… iii
KATA PENGANTAR……. iv
HALAMAN PERNYATAAN PERSETUJUAN PUBLIKASI……. v
ABSTRAK…… vi
ABSTRACT……. vii
DAFTAR ISI…… viii
DAFTAR GAMBAR…… x
DAFTAR TABEL……... xii
DAFTAR ISTILAH……. xiii
BAB 1.PENDAHULUAN…….. 1
1.1. Latar Belakang…….. 1
1.2. Perumusan Masalah…….. 2
1.3. Tujuan Penulisan……..... 2
1.4. Pembatasan Masalah……. 3
1.5. Metode Penelitian…….. 3
1.6. Sistematika Penulisan……. 3
BAB 2.LANDASAN TEORI……. 5
2.1. Prinsip Compliance Control…….. 5
2.1.1. Sensasi Kontak……..... 11
2.2. Newton Euler Equation…… 16
2.2.1. Recursive Newton-Euler Equation…….. 17
2.3. Motor Serial Servo……. 19
2.3.1. Dynamixel RX-24F……. 19
2.3.2. Protokol Serial Komunikasi…… 21
2.4. Motor DC gearbox 6volt……. 37
2.5. Mikrokontroler ATmega 128……..... 38
2.6. Perangkat Komunikasi (RS-485)…… 43

BAB 3.PERANCANGAN……... 45
3.1. Perancangan Perangkat Keras…….. 45
3.1.1. Sistem Mekanik Robot……. 45
3.1.2. Sistem Elektrik Robot……. 47
3.2. Forward Kinematik Menggunakan Homogeneous Transformation 48
3.3. Resolved Motion Rate Control Menggunakan Jacobian Invers…. 51
3.4. Newton-Euler Equation untuk Mancari Dinamika Sistem…… 55
3.5. Algoritma Komunikasi antara Mikrokontroler dengan Dynamixel. 58
3.6. Algoritma Compliance Control…….. 60
BAB 4.ANALISA…….. 62
4.1. Pengujian Position Control Tanpa Compliance Controller……. 62
4.1.1. Pengujian Tanpa Position Controller…… 62
4.1.2. Pengujian Menggunakan Position Control dengan Parameter Controller yang Optimal…….. 64
4.2. Pengujian Penambahan Compliance Control…….. 66
4.2.1. Pengujian Tanpa Kontak Luar (free motion)…… 66
4.2.2. Pengujian Dengan Kontak Luar (ada gaya ekternal)….. 68
BAB 5.KESIMPULAN…… 72
5.1. Kesimpulan……. 72
DAFTAR PUSTAKA……. 73
LAMPIRAN…….. 75

DAFTAR GAMBAR

Gambar 2.1. Konsep Compliance Control…… 6

Gambar 2.2. Konversi dari Gaya Eksternal ke Posisi Command…… 7

Gambar 2.3. Controller PD yang ditambah dengan Compliance Control… 7

Gambar 2.4. Blok Diagram Compliance Control yang Dirancang Pada Lengan Robot…….. 9

Gambar 2.5. Blok Diagram Motor Berdasarkan Disturbance Torsi…….. 12

Gambar 2.6. Kalkulasi Disturbance Torque didasarkan Pada Akselerasi… 13

Gambar 2.7. Kalkulasi Disturbance Torque dengan LPF…… 13

Gambar 2.8. Kalkulasi Disturbance Torsi Didasarkan Pada Kecepatan Angular (Disturbance Observer)……. 14

Gambar 2.9. Feedback dengan Disturbance Torque yang Diestimasi…….. 14

Gambar 2.10. Blok Diagram Equivalent Gambar 2.8…… 14

Gambar 2.11. Torsi Reaksi Observer…….. 16

Gambar 2.12. Model Friction…….. 16

Gambar 2.13. Ilistrasi Newto-Euler Vectors……... 17

Gambar 2.14. Relative Acceleration……..…… 18

Gambar 2.15. Dynamixel RX-24F…….. 20

Gambar 2.16 Pin Assignment Dynamixel RX-24F……. 21

Gambar 2.17. Wire Link Dynamixel RX-24F……. 21

Gambar 2.18. Aliran Paket Data dalam Komunikasi dengan Dynamixel… 21

Gambar 2.19. Koneksi Multidrop RX-24F…….. 22

Gambar 2.20 Toleransi Margin dan Slope…….. 29

Gambar 2.21. Goal Position Dynamixel RX-24F……... 31

Gambar 2.22. Algoritma Konfigurasi Awal Dynamixel RX-24F…….. 33

Gambar 2.23. Instruksi paket set ID…….. 34

Gambar 2.24. Instruksi paket set Status Return Level……... 34

Gambar 2.25. Konfigurasi compliance margin dan slope…….. 35

Gambar 2.26. Instruksi Paket Set Nilai Slope……. 36

Gambar 2.27. Instruksi Paket Set Nilai Margin…… 36

Gambar 2.28. Instruksi Paket Set Baudrate……. 37
Gambar 2.29. DC Gearmotor…….. 37
Gambar 2.30. ATmega 128……. 39
Gambar 2.31. Komunikasi Asynchronous…… 39
Gambar 2.32. Diagram Alir Komunikasi……. 43
Gambar 2.33. Rangkaian RS-485…… 44
Gambar 3.1. Lengan Robot…… 45
Gambar 3.2. Design Telapak Tangan Tobot…….. 46
Gambar 3.3. Gambar Asli Telapak Tangan Robot…….. 46
Gambar 3.4. Power Distribution Diagram……... 47
Gambar 3.5. Diagram Elektrik Tangan Robot……. 48
Gambar 3.6. Ilustrasi Koordinat Frames Lengan Robot…….. 49
Gambar 3.7. Ilustrasi Tangan Robot…….. 55
Gambar 3.8. Vektor Komponen Gaya Link 1…… 56
Gambar 3.9. Vektor Komponen Gaya Link 2…… 56
Gambar 3.10. algoritma protokol komunikasi antara mikrokontroller ATmega128 dengan dynamixel RX-24F…… 59
Gambar 3.21. Algoritma Compliance Control yang dirancang……. 60
Gambar 4.1. Nilai Posisi Referensi dan sensor Dynamixel Sebelum Penambahan Position Controller…… 63
Gambar 4.2. Nilai Error Posisi Sebelum Penambahan Position Controller. 63
Gambar 4.3. Nilai Posisi Referensi dan Sensor Dynamixel Setelah Penambahan Position Controller…….. 65
Gambar 4.4. Nilai error sudut setelah penambahan Position Controller. 65
Gambar 4.5. Nilai Torsi dari Sensor dan Dari Persamaan NewtonEuler.. 67
Gambar 4.6. Nilai Error Torsi Dari Sensor dan Persamaan Newton-Euler. 67
Gambar 4.7. Percobaan Compliance controller……. 69
Gambar 4.8. Nilai Posisi dari Sensor dan Referensi tanpa compliance control…… 69
Gambar 4.9. Error Posisi Dynamixel…….. 69
Gambar 4.10. Percobaan Compliance controller…… 70

Gambar 4.11. Nilai Posisi Referensi dan Command dengan Compliance Control…….. 71
Gambar 4.12. Error Posisi Dynamixel…… 71

DAFTAR TABEL

Tabel 2.1 Tabel Control RX-24F…….. 24
Tabel 2.2. Rumus Bautrate…….. 27
Tabel 2.3. Status Return Level……. 28
Tabel 2.4. Nilai Slope…….. 30
Tabel 2.5 Tabel Beban……. 32
Tabel 2.6. Instruksi Paket…… 32
Tabel 2.7 Register UCSRB…….. 40
Tabel 2.8 Deskripsi Register UCSRB…… 41
Tabel 2.9 Register UCSRA……. 41
Tabel 2.10 Deskripsi Register UCSRA…….. 41
Tabel 4.1 Parameter Compliance Control……. 68

DAFTAR ISTILAH

USART	Universal Synchronous Asynchronous Receiver Transmiter
RX	Receiver
TX	Transmitter
UBBR	Register 16 bi t pada mikrokontroler AVR yang berguna untuk menentukan kecepatan komunikasi USART
UDR	Register yang digunakan mikrokontroler AVR untuk bertukar data
UCSRA	Register yang menunjukkan status UART
RXC	Bit pada UCSRA yang mengindikasikan penerimaan data
TXC	Bit pada UCSRA yang mengindikasikan pengiriman data
UDRE	Bit pada UCSRA yang mengindikasikan register UDR kosong
UCSRB	Register yang bertujuan untuk menginisialisasi dan mengeset fungsi USART
RXEN	Bit pada UCSRB yang meng-enable pin RX
TXEN	Bit pada UCSRB yang meng-enable pin TX

BAB I
PENDAHULUAN

1.1. Latar Belakang

Pada saat ini teknologi robotika telah berkembang dengan pesat. Teknologi robot telah banyak digunakan dalam bidang industri untuk meringankan kerja manusia. Kemampuan robot untuk melakukan gerakan manusia sangat membantu dunia industri seperti industri mobil, proses pengelasan, perakitan, pemindahan dan banyak lagi. Salah satu contoh yang paling umum adalah manipulator yang berbentuk mirip tangan manusia dan dikendalikan oleh operator.

Disisi lain, sekarang sudah banyak juga dikembangkan robot yang memiliki intelejensi dan mampu berinteraksi dengan manusia. Misalnya robot pembantu untuk membantu meringankan pekerjaan rumah, robot animaloid sebagai teman bermain atau hiburan, robot penerima tamu, dan masih banyak lagi yang lainnya. Dengan semakin dekatnya interaksi robot dengan manusia, maka perlu adanya teknologi robot yang lebih bersahabat dan tidak membahayakan serta responsif terhadap lingkungan.

Untuk mendukung itu semua, dibutuhkan intelegensi robot yang tinggi yang mampu respon terhadap tungkah laku benda-benda sekitarnya dan perubahan lingkungan, misalnya sentuhan manusia, gerakan manusia, suara, perubahan suhu, dan tekanan. Oleh karena itu, sangat dibutuhkan sensor-sensor dengan tingkat presisi dan ketelitian yang tinggi yang dapat mendeteksi lingkungan sekitarnya. Berbagai macam sensor pun muncul dengan kegunaan yang bervariasi.

Robot penerima tamu merupakan salah satu contoh robot *humanoid* yang didesign untuk bertingkah laku seperti penerima tamu layaknya manusia. Disebut robot *humanoid* karena robot penerima tamu umumnya berbentuk seperti manusia, memiliki kepala, tangan, badan, dan kaki. Tangan banyak berinterkasi dengan manusia dan lingkungan secara langsung misalnya untuk berjabat tangan, mengambil minuman, ataupun memegang benda-benda lainya. Untuk itu dibutuhkan suatu teknologi atau metode yang mampu membuat tangan robot dapat berinteraksi secara langsung, responsif, tidak mebahayakan, dan lebih

bersahabat. Salah satu metode yang banyak digunakan agar robot dapat berinteraksi dan respon terhadap gaya dan tekanan dari luar adalah dengan menerapkan metode *compliance control.*

Strategi *compliance control* pada dasarnya dibagi menjadi dua group, yaitu *active compliance* dan *passive compliance. Passive compliance* menggunakan divais mekanik yang fleksibel yang ditempatkan pada ujung robot. Berhubungan dengan tugas yang di susun, divais ini mengatur posisi obyek yang dimanipulasi ketika sebuah gaya interaksi diterapkan. Sedangkan *active compliance* adalah suatu metode yang didasarkan pada divais aktif, seperti *Force/Torque (F/T)* sensors untuk mengukur interaksi dengan robot. Metode *compliance control* yang banyak digunakan untuk sekarang ini adalah *active compliance*.

Compliance control merupakan kombinasi antara *motion control* dan *force control.* Dengan menerapkan *Resolved Motion Rate Control (RMRC)*, tangan robot dapat bergerak sesuai instruksi yang di perintahkan dan setiap waktu, posisi, kecepatan, serta gaya aktual pada setiap joint dapat terekam melalui sensor. Dengan menggunakan *Neuton-Euler equation* tentang dinamika maka bisa didapatkan persamaan gaya (force) ditiap titik gerakan. Kinematik mentransformasikan koordinat kartesius menjadi koordinat sudut sedangkan inverse kinematik sebaliknya. Dengan menerapkan keduanya maka dengan lebih mudah *motion control* dapat diterapkan.

1.2. Perumusan Masalah

Masalah yang akan dibahas secara khusus adalah tentang perancangan tangan robot dan mengimplementasikan alghoritma *compliance control* untuk membuat robot dapat *compliance* terhadap lingkungan. Skripsi ini bertujuan untuk menerapkan *motion control* dengan *Resolved Motion Rate Control (RMRC)* dan *force control* dengan metode *Dynamic Neuton-Euler Equation* pada tangan robot serta menguji strategi *compliance control* .

1.3. Tujuan Penulisan

Merancang tangan robot dan mengimplementasikan strategi *compliance control* dengan mengkombinasikan *motion control* dengan implemantasi

Resolved Motion Rate Control (RMRC) dan *force control* dengan metode *Dynamic Neuton-Euler Equation* pada tangan robot.

1.4. Pembatasan Masalah

Pada skripsi ini, kerangka tangan robot yang digunakan terbuat dari bahan acrylic 3 mm yang cukup ringan dan tangan digantung pada kerangka bahu buatan. *Motion control* dan *force control* yang diterapkan masih dalam ruang lingkup bidang 2 dimensi. Sensor-sensor yang digunakan antara lain adalah resistif force sensor, current sensor, serta feedback posisi, kecepatan, dan load yang ada pada dynamixel RX-24F.

1.5. Metode Penelitian

Metode penulisan yang dilakukan dalam membuat skripsi ini meliputi

a. Studi Literatur, yaitu dengan mencari referensi tentang unsur-unsur pembentuk tangan robot, metode-metode *compliance control*, dan sitem pengujian algoritma *compliance control*.
b. Pendekatan diskusi dengan pembimbing skripsi
   Mendiskusikan metode serta perancangan dengan pembimbing skripsi.
c. Eksperimen, yaitu dengan menerapkan algoritma yang telah didapat pada sistem fisik dan menguji algoritma yang sudah diterapkan.
d. Membuat kesimpulan
   Dari hasil analisa data, dapat menarik kesimpulan sehingga dapat melihat performa compliance control yang telah dirancang.

1.6. Sistematika Penulisan

a. Bab I akan menjelaskan tentang pendahuluan, yaitu berupa latar belakang, perumusan masalah, tujuan penelitian, pembatasan masalah, metodologi penelitan dan sistematika penulisan.
b. Bab II akan menjelaskan landasan teori yang meliputi seluruh prinsip compliance control, actuator, sensor dan mikrokontroller.

c. Bab III akan menjelaskan tentang perancangan *compliance control* meliputi design hardware dan implementasi *compliance control* pada tangan robot dengan metode *Dynamics Newton-Euler Equation*.
d. Bab IV akan menjelaskan tentang pengujian yaitu berupa pengujian hasil implementasi *compliance control* yang sudah dirancang.
e. Bab V akan menjelaskan tentang kesimpulan dari hasil penelitian.

BAB II
LANDASAN TEORI

Metode compliance control dibutuhkan untuk membangun robot yang mampu berinteraksi dengan manusia tanpa membahayakan atau melukai. Robot diharapkan untuk memainkan peran sebagai partner manusia. . Pada bab ini akan dibahas mengenai pengenalan prinsip compliance control secara umum, Newton Euler Dynamic, dan aktuator yang digunakan dalam rancang bangun compliance control, yaitu dynamixel RX-24F.

2.1. Prinsip Compliance Control

Compliance control adalah sebuah sistem control dengan kompensasi trajectory sehingga gaya eksternal (*eksternal force*) mungkin diikuti. Kebutuhan untuk kontrol gaya meningkat untuk kebutuhan performa gerakan yang tangkas. Sebagian besar, motion controller menjabarkan/menjalankan kumpulan input ke actuator berdasarkan referensi motion. Output motion akan menjadi posisi dan/gaya. Untuk diterapkan pada kehidupan manusia, antara kontrol posisi dan gaya seharusnya di pertunjukan secara simultan, sehingga robot tidak akan membahayakan/melukai manusia. Akan tetapi, ideal kontrol posisi bertingkah laku deviasi posisi melawan deviasi gaya. Sedangkan, kontrol gaya ideal bertingkah laku deviasi gaya melawan deviasi posisi. Didalam compliance control, harus ada hubungan antara posisi dan gaya[1].

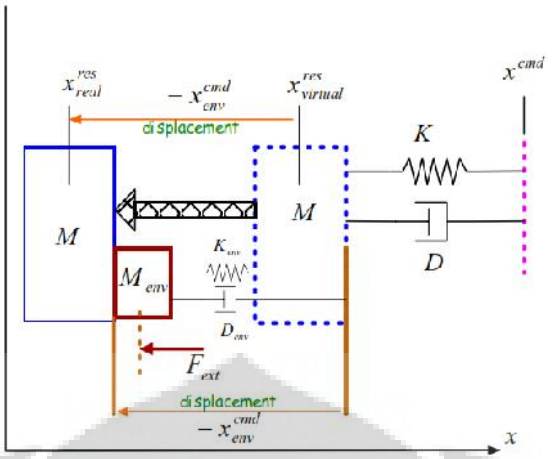
Konsep compliance control di ilustrasikan pada gambar 2.1. Anggap robot diperintahkan untuk pencapai x^{cmd}, tetapi karena dinamik robot, robot masih berada di $x^{res}_{virtual}$. Dan formulasinya dapat ditulis sebagai berikut.

$$M\ddot{x}^{res}_{virtual} + \mathrm{K}(x^{res}_{virtual} - x^{cmd}) + \mathrm{D}(\dot{x}^{res}_{virtual} - \dot{x}^{cmd}) = 0 \qquad 2.1)$$

Kemudian

$$\ddot{x}^{res}_{virtual} = \frac{1}{M}\{\mathrm{K}(x^{cmd} - x^{res}_{virtual}) + \mathrm{D}(\dot{x}^{cmd} - \dot{x}^{res}_{virtual})\}$$
$$= \mathrm{K_p}(x^{cmd} - x^{res}_{virtual}) + \mathrm{K_v}(\dot{x}^{cmd} - \dot{x}^{res}_{virtual}) \qquad 2.2)$$

$$x^{res}_{virtual} = \frac{K_v s + K_p}{s^2 + K_v s + K_p} x^{cmd} \quad [1] \qquad 2.3)$$



Gambar 2.1. Konsep Compliance Control[1]

Akan tetapi, karena virtual obyek M_{env} , sistem akan mempunyai pergeseran virtual sebanyak x_{env}^{cmd}. Gaya eksternal f_{ext} pada virtual obyek direpresentasikan dengan gaya yang diberikan untuk obyek ini atau gaya reaksi sebagai motion robot yang menekan virtual obyek M_{env} . Disini, robot menjadi compliance. Sebagian besar compliance control adalah sebuah kontrol sistem dengan kompensasi trajectory sehingga eksternal gaya mungkin diikuti. Demikian, kontak stabil dengan lingkungan adalah mampu untuk merealisasikan dengan skema kontrol ini menjadi mungkin untuk memberikan robot flesibilitas dan cooperative[1]. Sebagai hasilnya, posisi robot actual akan menjadi,

$$x_{real}^{res} = x_{virtual}^{res} - x_{env}^{cmd} \qquad 2.4)$$

Sehingga,

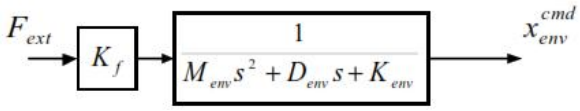
$$x_{real}^{res} = \frac{K_v s + K_p}{s^2 + K_v s + K_p} x^{cmd} - x_{env}^{cmd}$$

$$= \frac{K_v s + K_p}{s^2 + K_v s + K_p} x^{cmd} - \frac{s^2 + K_v s + K_p}{s^2 + K_v s + K_p} x_{env}^{cmd}$$

$$(s^2 + K_v s + K_p) x_{real}^{res} = (K_v s + K_p) x^{cmd} - (s^2 + K_v s + K_p) x_{env}^{cmd} \qquad 2.5)$$

$$x_{real}^{res} = K_p(x^{cmd} - x_{real}^{res} - x_{env}^{cmd}) + K_v(\dot{x}^{cmd} - \dot{x}_{real}^{res} - \dot{x}_{env}^{cmd}) - \ddot{x}_{env}^{cmd} \; [1] \; 2.6)$$

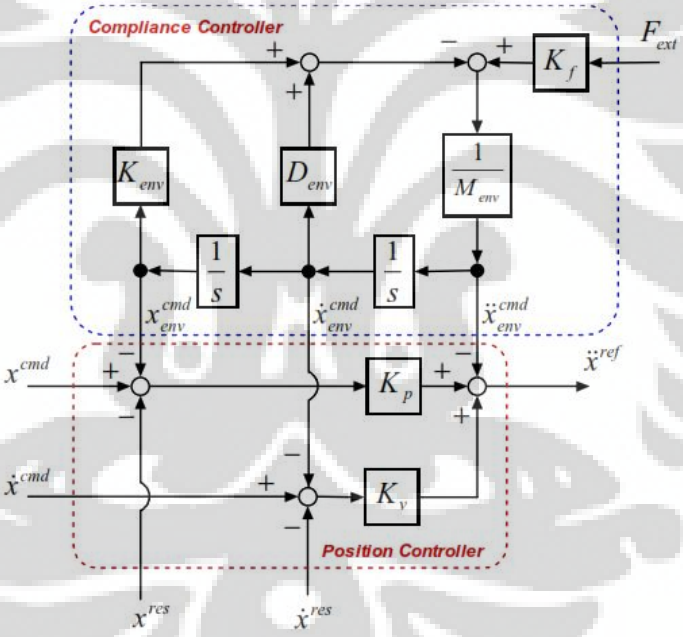
Pergeseran dapat dimodelkan dimodelkan dengan workspace impedance virtual sebagaimana ditunjukkan pada gambar 2.2 diturunkan dari struktur mekanikpada gambar 2.1. dengan formulasi (2.7). Parameter workspace virtual ini dapat diubah sebaik mungkin untuk menyediakan responinput gaya yang diharapkan.



Gambar 2.2 Konversi dari gaya eksternal ke posisi command[1]

$$x_{env}^{cmd} = \frac{K_f}{M_{env}s^2 + D_{env}s + K_{env}} F_{ext} \quad (2.7)$$

Dalam mendapatkan compliance control, sebuah controller posisi PD konvensional akan diperbaiki kedalam structure control yang ditunjukkan pada gambar 2.3. Prakteknya, sebuah compliance controller meningkatkan controller utama (PD controller) untuk menjadi compliance dengan kondisi pasti sehingga sistem bekerja dengan baik. Kondisi pasti tidak dibatasi untuk gaya, tetapi juga momentum, kecepatan, pergeseran, atau apapun pada lingkungan sehingga mungkin menginduksi performa motion[1].



Gambar 2.3 Controller PD yang ditambah dengan compliance controller[1]

Respon sistem keseluruhan dapat di amati dengan menurunkan persamaan (2.4) ke dalam,

$$x_{real}^{res} = \frac{K_v s + K_p}{s^2 + K_v s + K_p} x^{cmd} - \frac{1}{M_{env}s^2 + D_{env}s + K_{env}} K_f F_{ext} \quad (2.8)$$

$$= \frac{K_v s + K_p}{s^2 + K_v s + K_p} x^{cmd} - \frac{\frac{K_{env}}{M_{env}}}{s^2 + \frac{D_{env}}{M_{env}}s + \frac{K_{env}}{M_{env}}} \frac{K_f}{K_{env}} F_{ext} \quad (2.9)$$

$$= \frac{2\delta_p\omega_p s+{\omega_p}^2}{s^2+2\delta_p\omega_p s+{\omega_p}^2} x^{cmd} - \frac{{\omega_f}^2}{s^2+2\delta_f\omega_f s+{\omega_f}^2} C_f F_{ext} \text{ [1]} \quad 2.10)$$

Dimana respon sistem dapat dirubah secara acak didasarkan pada frekuensi yang diharapkan dan dan damping rasio sebagaimana,

$$\omega_p = \sqrt{K_p} \quad 2.11)$$

$$\varsigma_p = \frac{K_v}{2\sqrt{K_p}} \quad 2.12)$$

$$\omega_f = \sqrt{\frac{K_{env}}{W_{env}}} \quad 2.13)$$

$$\varsigma_p = \frac{D_{env}}{2\sqrt{M_{env}K_{env}}} \quad 2.14)$$

$$K_f = \frac{K_f}{K_{env}} \quad 2.15)$$

Didasarkan pada pendekatan second order, dapat dengan mudah memutuskan parameter itu dengan respon yang diharapkan sebagai,

$$T_p = \frac{\pi}{\omega_n\sqrt{1-\varsigma^2}} = \frac{\pi}{\omega_d} \quad 2.16)$$

$$\%OS = e^{\left(\varsigma\pi/\sqrt{1-\varsigma^2}\right)} x100\% \quad 2.17)$$

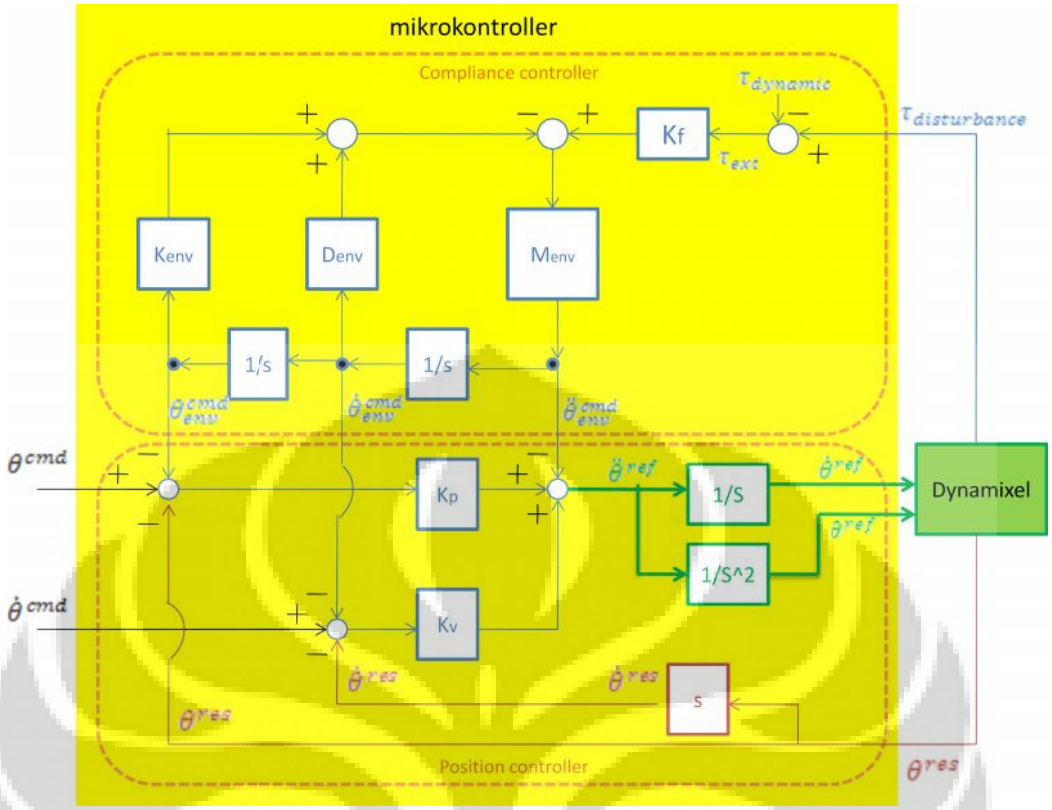
$$T_s = 4/\varsigma\omega_n \quad 2.18)$$

$$T_r = 1.8/\omega_n \quad 2.19)$$

Dimana T_p , T_s , dan T_r adalah peak time, settling time, dan rise time respectively[1].

- **Compliance Control yang Dirancang**

Konsep dasar compliance control yang diterapkan pada sistem tangan robot adalah dengan memanfaatkan feedback posisi, kecepatan, dan torsi dari dynamixel serta current sensor pada gripper. Untuk mendapatkan persamaan dynamic sistem digunakan persamaan Neuton-Euler dengan memperhitungkan pengaruh gravitasi yang bekerja pada sistem. Blok Diagram compliance control yang sudah dirancang untuk sistem lengan robot adalah sebagai berikut.



Gambar 2.4 Blok Diagram Compliance Control yang Dirancang Pada Lengan Robot

Dari gambar diatas dapat kita lihat bahwa trajectory yang diharap akan masuk ke dalam controller. Dengan adanya compliance control (kombinasi position control dan force control), maka torsi/gaya eksternal akan mengkompensasi/mengubah trajectory sistem yang memungkinkan torsi/gaya eksternal tersebut diikuti. Keluaran mikrokontroller berupa posisi sudut θ^{res} dan kecepatan sudut $\dot{\theta}^{ref}$ yang kemudian akan dikirim ke dynamixel melalui komunikasi serial. Kemudian dynamixel akan bekerja dan memberikan feedback berupa disturbance torque $\tau_{disturbance}$, kecepatan angular aktual $\dot{\theta}^{res}$, dan posisi aktual θ^{res} yang diambil dari nilai sensor yang ada pada dynamixel. Compliance control secara umum dapat dituliskan sebagai berikut.

$$compliance\ control = Position\ control - torsi\ atau\ gaya\ eksternal\ (\tau_{ext})$$

Output dari compliance control dapat direpresentasikan sebagai percepatan angular $\ddot{\theta}^{ref}$ yang kemudian di integralkan menjadi kecepatan angular $\dot{\theta}^{ref}$ dan posisi angular θ^{ref}. Position control secara umum direpresentasikan

$$\ddot{\theta}^{ref} = K_p(\theta^{cmd} - \theta^{real}) + K_v(\dot{\theta}^{cmd} - \dot{\theta}^{real}) \quad 2.20)$$

Untuk torsi disturbance (load), persamaanya dapat direpresentasikan

$$\tau_{disturbance} = \tau_{dynamic} + \tau_{ext} + (F + D\dot{\theta}) \quad 2.21)$$

Load torque adalah penjumlahan dari torsi dynamic (inersia) $\tau_{dynamic}$, torsi eksternal τ_{ext} , dan torsi friction $(F + D\dot{\theta})$. Torsi inersia diturunkan dari Newton-Euler equation. Eksternal torsi ada karena kontak tugas dengan lingkungan. Torsi friction adalah penjumlahan coulomb dan torsi viskositas. Pada kasus ini torsi friction dianggap nol karena pengaruh yang kecil. Akan tetapi untuk mendapatkan hasil yang lebih akurat memang perlu melibatkan torsi friction. Dengan menghilangkan komponen torsi friction maka persamaan torsi disturbance dapat disederhanakan menjadi

$$\tau_{disturbance} = \tau_{dynamic} + \tau_{ext} \quad 2.22)$$

Maka kita dapat memperoleh torsi eksternal dengan mengurangi torsi disturbance (hasil dari sensor) dengan torsi dynamic (hasil perhitungan Neuton Euler).

$$\tau_{ext} = \tau_{disturbance} - \tau_{dynamic} \quad 2.23)$$

Torsi eksternal ini dijabarkan dalam bentuk orde dua agar mendapatkan respon sistem yang lebih baik. Maka torsi ekternal dapat direpresentasikan dalam bentuk

$$\tau_{ext} = (M_{env}s^2 + D_{env}s + K_{env})x_{env}^{cmd} \quad 2.24)$$

$$= M_{env}\ddot{\theta}_{env}^{cmd} + D_{env}\dot{\theta}_{env}^{cmd} + K_{env}\theta_{env}^{cmd} \quad 2.25)$$

Sehingga persamaan compliance control dapat direpresentasikan kembali ke dalam bentuk

$$compliance\ control = Position\ control - torsi\ atau\ gaya\ eksternal\ \tau_{ext}$$

$$\ddot{\theta}^{ref} = K_p(\theta^{cmd} - \theta^{real} - K_{env}\theta_{env}^{cmd}) + K_v(\dot{\theta}^{cmd} - \dot{\theta}^{real} - D_{env}\dot{\theta}_{env}^{cmd}) - M_{env}\ddot{\theta}_{env}^{cmd} \quad 2.26)$$

2.1.1. Sensasi Kontak

Secara umum, sensasi kontak di realisasikan dengan menggunakan force sensor. Akan tetapi, gaya eksternal yang dirasakan oleh force sensor dideteksihanya pada posisi dimana force sensor diletakkan. Itu akan menjadi masalah untuk lingkungan yang kompleks jika robot bertubrukan dengan lingkungan tidak mengenai force sensor dimana force sensor itu diletakkan. Pada kenyataanya, gaya eksternal mungkin menstimulasi tingkah laku actuator. Disini, gaya eksternal di perlihatkan sebagai bagian torsi gangguan pada actuator. Secara umum, untuk motor servo DC sebagai actuator, persamaan dinamik motor di ekspresikan sebagai,

$$J\ddot{\theta} = \tau_m - \tau_i \quad [1] \qquad 2.27)$$

Dimana $\tau_m, \tau_i, J, \theta$ adalah torsi yang diberikan, load (disturbance) torque, inersia, dan pergeseran/perpindahan angular motor. Load torque representasikan sebagai,

$$\tau_i = \tau_{int} + \tau_{ext} + \left(F + D\dot{\theta}\right) \quad [1] \qquad 2.28)$$

Load torque adalah penjumlahan dari torsi inersia τ_{int}, torsi eksternal τ_{ext} , dan torsi friction $\left(F + D\dot{\theta}\right)$. Torsi inersia diturunkan dari Newton-Euler equation. Eksternal torsi ada karena kontak tugas dengan lingkungan. Torsi friction adalah penjumlahan coulomb dan torsi viskositas.

Untuk motor servo dynamixel RX-24F, perumusan torsi dynamixel sudah di set dari pabrik yang direprsentasikan dengan nilai 0-1023 dengan maksimal torsi 26 kgf.cm (pada kondisi 12V, 2.4A). Sedangkan torsi load (beban) dapat diambil dari data sensor torsi yang ada pada dynamixel yang direpresentasikan dengan nilai 0-1023. Penjelasan lebih detail tentang motor servo dynamixel akan dijelaskan pada BAB berikutnya.

Untuk motor DC, torsi motor adalah produk koefisien torsi yang berkoresponden dengan flux magnetic dengan arus torsi.

$$\tau_m = K_f I_a = K_f I_a^{ref} \qquad 2.29)$$

Dimana K_t, I_a adalah koefisien torsi dan arus torsi.

Subtitusikan persamaan (2.22) dan (2.23) kedalam (4.21) menghasilkan,

$$J\ddot{\theta} = K_f I_a^{ref} - (\tau_{int} + \tau_{ext} + \left(F + D\dot{\theta}\right)) \quad [1] \qquad 2.30)$$

Catatan bahwa parameter pada persamaan (2.24) hanya inersia *J* dan koefisien torsi K_t. Inersia akan berubah berdasarkan konfigurasi mekanik dari sistem motion.

$$J = J_n + \Delta J \qquad 2.31)$$

Variasi torsi karena perubahan inersia adalah $J\ddot{\theta}$. Koefisien torsi akan bervariasi berdasarkan posisi rotor motor elektrik karena irregular distribution fluk magnetic pada permukaan motor.

$$K = K_{tn} + \Delta K_t \qquad 2.32)$$

Ripple torsi karena ruang harmonic adalah $-\Delta K_t I_a^{ref}$

Disini, total disturbance torque τ_{dis} direpresentasikan sebagai,

$$\tau_{dis} = \tau_i + \Delta J\ddot{\theta} - \Delta K_t I_a^{ref} \qquad 2.33)$$

$$= I_a^{ref} + \tau_{int} + \tau_{ext} + \left(F + D\dot{\theta}\right) + (J - J_n)\ddot{\theta} + (K_{tn} - K_t)I_a^{ref} \quad [1] \qquad 2.34)$$

Persamaan dinamik dasar di bentuk kedalam,

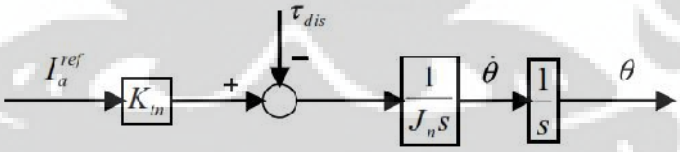
$$(J_n + \Delta J)\ddot{\theta} = (K_{tn} - K_t)I_a^{ref} - \tau_i$$

$$J_n\ddot{\theta} = K_t I_a^{ref} - (\tau_i + \Delta J\ddot{\theta} - \Delta K_t I_a^{ref})$$

$$J_n\ddot{\theta} = K_t I_a^{ref} - \tau_{dis} \quad [1] \qquad 2.35)$$

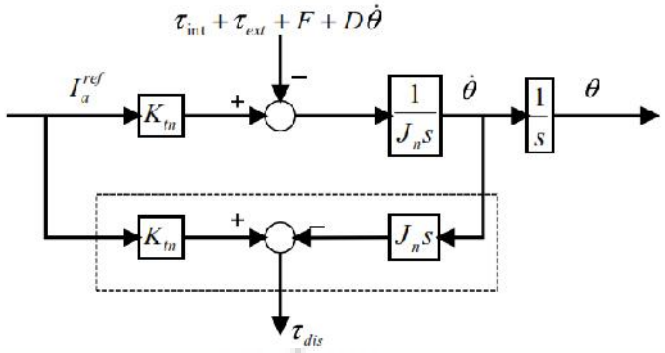
Blok diagram motor didasarkan pada disturbance torsi ditunjukkan pada gambar 2.5 dan persamaan 2.29 dapat di bentuk kedalam,

$$\tau_{dis} = K_t I_a^{ref} - J_n\ddot{\theta} \qquad 2.36)$$



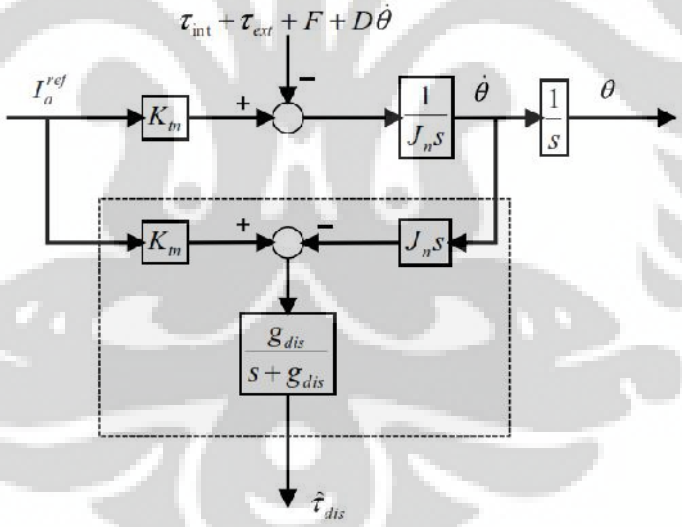
Gambar 2.5 Blok Diagram Motor DC Berdasarkan Disturbance Torsi[1]

Sisi kiri persamaan (2.29) adalah penjumlahan faktor yang tidak diketahui, misalnya load yang tidak terprediksi dan variasi parameter yang tidak diketahui, tetapi, persamaan sisi kanan diketahui. Oleh karena itu, disturbance dikalkulasi sebagai yang ditunjukkan pada gambar 2.6.

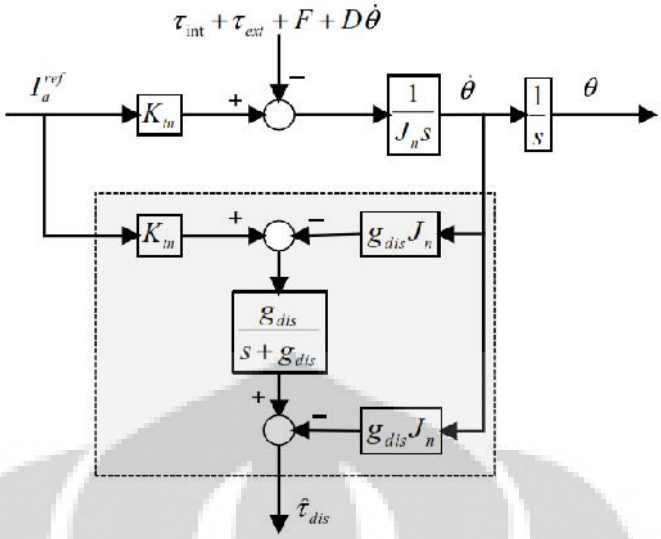


Gambar 2.6. Kalkulasi Disturbance Torque Didasarkan pada Akselerasi[1]

Karena gangguan yang di estimasi mungkin berubah tiba-tiba, dan akselerasi untuk estimasi diturunkan dengan proses differensial biasa, maka low-pass filter (LPF) digunakan untuk merelaksasi hasil estimasi yang ditunjukkan pada gambar 2.7. Disini g_{dis} mendenotasikan sebuah frekuensi cut off angular first order LPF. Jika g_{dis} cukup besar, maka disturbance torque yang diestimasi hamper mirip dengan yang sebenarnya. Model ekuivalen gambar 2.6 ditunjukkan pada gambar 2.8 yang biasanya disebut *disturbance observer*.

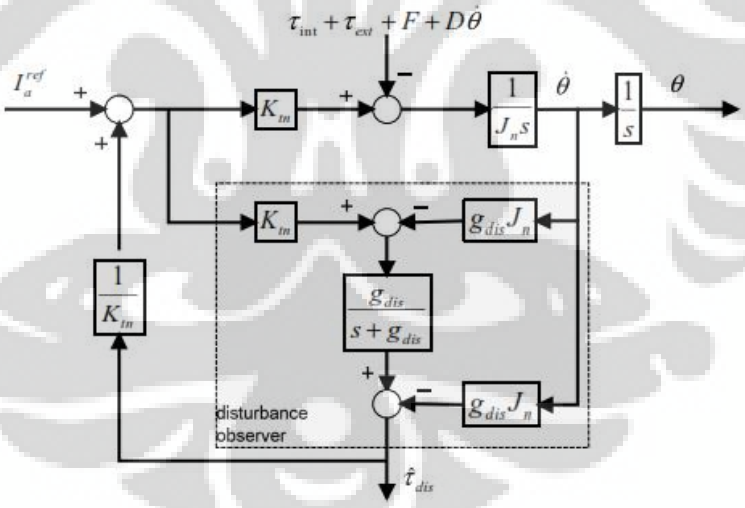


Gambar 2.7 Kalkulasi Disturbance Torque dengan LPF[1]

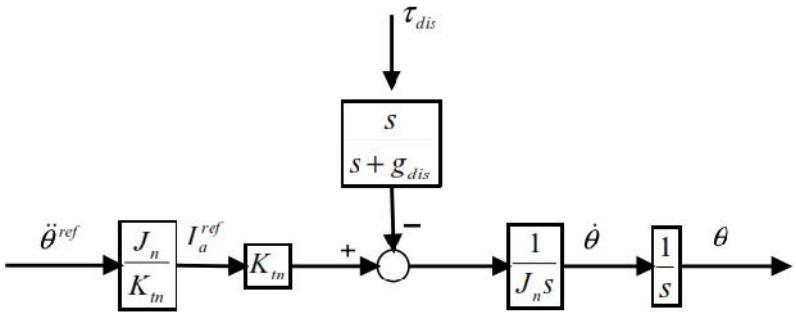


Gambar 2.8 Kalkulasi Disturbance Torsi Didasarkan Pada Kecepatan Angular (Disturbance Observer[1]

Secara umum, disturbace torsi yang diestimasi dapat digunakan sebagai feedback untuk menahan gangguan pada sistem yang ditunjukkan pada gambar 2.9 Disini, block diagram ekuivalen di tunjukkan pada gambar 2.10.



Gambar 2.9 Feedback dengan Disturbance Torque yang Diestimasi[1]



Gambar 2.10 Blok Diagram Equivalent Gambar 2.8[1]

Kenyataannya, disturbance torque yang diestimasi tidak hanya efektif untuk kompensasi disturbance tetapi juga parameter identifikasi pada sistem mekanik. Dengan menggunakan dua disturbance observer independen dengan asumsi bahwa efek disturbance torque ditekan dengan baik pada motion. Identifikasi dinamik dapat dicapai. Disini, satu disturbance observer digunakan untuk merealisasikan kontroler robust motion dan yang lainnya digunakan untuk mengestimasi parameter dinamik motion.

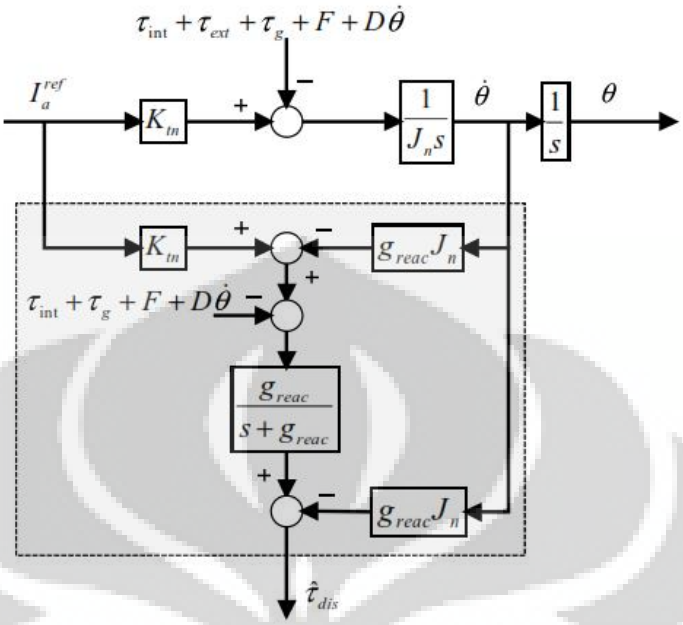
Secara umum, dinamik sistem mekanik berisi component inersia,komponen potensial, dan komponen gaya eksternal. Persamaan 2.37 memberikan penjumlahan element diatas. Hal ini mungkin untuk mendeteksi komponen pulsa torsi dengan distribusi fluk magnetic pada motor. Dengan mengurangi komponen pulsa torsi dari output disturbance observer, penjumlahan komponen inersia dan komponen potensial dapat dikalkulasi. Dua komponen ini juga diturunkan dari Neuton-Euler equation. Tes kecepatan konstan dan test motion yang dipercepat pada situasi tersebut menghasilkan parameters dinamik. Secara umum, kecepatan konstan mengetes estimasi gravitasi dan friction and tes motion yang dipercepat mengestimasi inersia.

Output disturbance observer hanya efek friction pada motion kecepatan angular konstan. Ciri-ciri ini membuat mungkin untuk mengidentifikasi efek friction pada sistem mekanik. Disini diasumsikan bahwa efek friction diketahui sebelumnya oleh proses identifikasi di atas. Dengan mengimplementasikan motion angular yang dipercepat, parameter-parameter sistem K_{tn} dan J_n diatur pada design observer sehingga tertutup pada nilai actual, berturut-turut. Ketika model dinamik sistem mekanik di identifikasi, mungkin untuk mengestimasi torsi reaksi sistem. Torsi reaksi yang diestimasi ditunjukkan sebagai berikut,

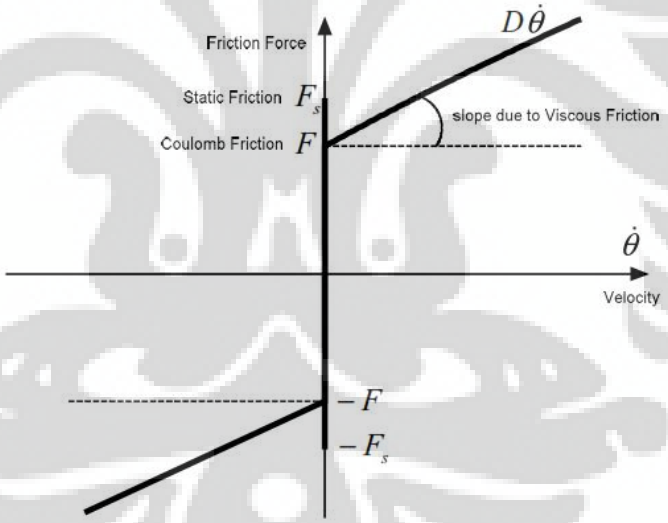
$$\hat{\tau}_{reac} = \frac{g_{reac}}{s+g_{reac}}(K_{tn}I_a^{ref} + g_{reac}J_n\dot{\theta} - \tau_{int} - \tau_g - F - D\theta) - \dot{g}_{reac}J_n\dot{\theta} \quad [1]2.37)$$

Process identifikasi torsi eksternal di ilustrasikan pada gambar 2.11, yang mana disebut torsi reaksi observer. Disini torsi reaksi observer dapat mengestimasi torsi eksternal motor tanpa sensor. Tetapi, karena efek friction adalah nonlinear disturbance sebagaimana yang ditunjukkan pada gambar 2.12,

identifikasi friction sangat sulit. Khususnya, friction static Fs adalah masalah yang serius untuk realisasi control gaya sensor-less.



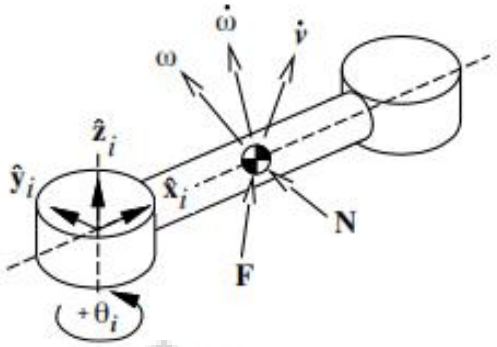
Gambar 2.11 Torsi Reaksi Observer[1]



Gambar 2.12 Model Friction[1]

2.2. Neuton Euler Equation

Metode Newton-Euler equation didasarkan pada persamaan hukum 2 Newton tentang gerak (motion) dan persamaan Euler tentang momen/gaya anguler[11].



Gambar 2.13. Ilustrasi Newton-Euler vectors[11]

Newton's Equation :

$$F = \frac{d}{dt}[R_i(m_i v_i)] = R_i(m_i \dot{v}_i) + \dot{R}_i(m_i v_i) \quad 2.38)$$
$$= R_i[(m_i \dot{v}_i) + (\omega_i \, x \, m_i v_i)] \quad [11]$$

Euler's Equation :

$$N = \frac{d}{dt}[R_i(M_i \omega_i)] \quad 2.39)$$
$$= R_i[M_i \omega_i + (\omega_i \, x \, m_i v_i)] \quad [11]$$

Dimana F dan N adalah gaya dan vektor torsi pada link i yang ditulis pada inertial koordinat. R_i adalah matriks rotasi yang menghubungkan frame i pada frame inersia, dan ω_i total kecepatan anguler link i ynag ditulis pada link i koordinat[11].

Jika F dan N ditulis pada frame koordinat lokal untuk link i, maka

$$\begin{bmatrix} mI_3 & 0 \\ 0 & M \end{bmatrix} \begin{bmatrix} \dot{v} \\ \dot{\omega} \end{bmatrix} + \begin{bmatrix} \omega \, x \, mv \\ \dot{\omega} \end{bmatrix} = \begin{bmatrix} F \\ N \end{bmatrix} = W \quad 2.40)$$

W adalah solusi yang berisi gaya dan torsi yang beraksi pada link i yang ditulis didalam link i koordinat. Jika kita dapat menjumlahkan semua state motion $(\omega, \dot{\omega}, \dot{v})$, maka kita dapat menghitung total load. W, yang beraksi pada pusat massa dan menentukan persamaan motion untuk link i[11].

2.2.1. Recursive Neuton-Euler Equation

Propagasi state motion $(\omega, \dot{\omega}, \dot{v})_i$ dari frame i ke frame (i+1) dapat dinyatakan dalam bentuk persamaan :

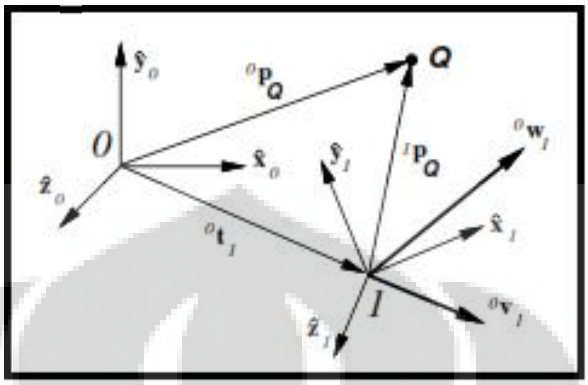
a. Anguler Velocity : $\boldsymbol{\omega}$

- Revolute : ${}^{i+1}\boldsymbol{\omega}_{i+1} = {}_{i+1}R_i \, {}^{i}\boldsymbol{\omega}_i + \dot{\theta}_{i+1} \hat{\boldsymbol{z}}_{i+1}$ 2.41)
- Prismatic : ${}^{i+1}\boldsymbol{\omega}_{i+1} = {}_{i+1}R_i \, {}^{i}\boldsymbol{\omega}_i$ 2.42)

b. Angular acceleration : $\dot{\boldsymbol{\omega}}$

- Revolute : ${}^{i+1}\dot{\boldsymbol{\omega}}_{i+1} = {}_{i+1}R_i\, {}^{i}\dot{\boldsymbol{\omega}}_i + ({}_{i+1}R_i\, {}^{i}\boldsymbol{\omega}_i \times \dot{\theta}_{i+1}\, \hat{\boldsymbol{z}}_{i+1}) + \ddot{\theta}_{i+1}\, \hat{\boldsymbol{z}}_{i+1}$ 2.43)
- Prismatic : ${}^{i+1}\dot{\boldsymbol{\omega}}_{i+1} = {}_{i+1}R_i\, {}^{i}\dot{\boldsymbol{\omega}}_i$ 2.44)

c. Linear Acceleration: $\dot{\boldsymbol{v}}$



Gambar 2.14. Relative Acceleration[11]

$${}^{0}\mathbf{P}_Q = {}_0R_1\, {}^{1}\mathbf{p}_Q + {}^{0}\mathbf{t}_1$$

$${}^{0}\mathbf{v}_Q = {}_0R_1\, {}^{1}\dot{\boldsymbol{p}}_Q + ({}^{0}\boldsymbol{\omega}_1 \times {}_0R_1\, {}^{1}\mathbf{p}_Q) + {}^{0}\mathbf{v}_1$$

$${}^{0}\dot{\boldsymbol{v}}_Q = \frac{d}{dt}[\, {}_0R_1\, {}^{1}\dot{\boldsymbol{p}}_Q\,] + ({}^{0}\dot{\boldsymbol{\omega}}_1 \times {}_0R_1\, {}^{1}\mathbf{p}_Q) + ({}^{0}\boldsymbol{\omega}_1 \times \frac{d}{dt}[\, {}_0R_1\, {}^{1}\mathbf{p}_Q\,]\,) + {}^{0}\dot{\boldsymbol{v}}_1$$

$${}^{0}\dot{\boldsymbol{v}}_Q = {}_0R_1\, {}^{1}\ddot{\boldsymbol{p}}_Q + ({}^{0}\boldsymbol{\omega}_1 \times {}_0R_1\, {}^{1}\dot{\boldsymbol{p}}_Q) + ({}^{0}\dot{\boldsymbol{\omega}}_1 \times {}_0R_1\, {}^{1}\mathbf{p}_Q\,) + ({}^{0}\boldsymbol{\omega}_1 \times {}_0R_1\, {}^{1}\dot{\boldsymbol{p}}_Q) \quad +$$

$$({}^{0}\boldsymbol{\omega}_1 \times {}^{0}\boldsymbol{\omega}_1 \times {}_0R_1\, {}^{1}\mathbf{p}_Q) + {}^{0}\dot{\boldsymbol{v}}_1 \qquad 2.45)$$

Sekarang subtitusikan :

frame 0 <=> frame (i-1)

frame 1 <=> frame (i)

frame 2 <=> frame (i+1)

$${}^{i+1}\dot{\boldsymbol{v}}_{i+1} = {}_{i+1}R_{i-1} + [\, {}_{i-1}R_i\, {}^{i}\ddot{\boldsymbol{p}}_{i+1} + 2({}^{i-1}\boldsymbol{\omega}_i \times {}_{i-1}R_1\, {}^{i}\dot{\boldsymbol{p}}_{i+1}\,) + ({}^{i-1}\dot{\boldsymbol{\omega}}_i \times {}_{i-1}R_i \quad {}^{i}\dot{\boldsymbol{p}}_{i+1}) + ({}^{i-1}\boldsymbol{\omega}_i \times {}^{i-1}\boldsymbol{\omega}_i \times {}_{i-1}R_1\, {}^{i}\mathbf{p}_{i+1}) + {}^{i}\dot{\boldsymbol{v}}_i\,]$$

- Revolute : ${}^{i}\mathbf{p}_{i+1}$=const, ${}^{i}\dot{\boldsymbol{p}}_{i+1} = {}^{i}\ddot{\boldsymbol{p}}_{i+1} = 0$

$${}^{i+1}\dot{\boldsymbol{v}}_{i+1} = {}_{i+1}R_i\, [\, {}^{i}\dot{\boldsymbol{v}}_i + ({}^{i}\dot{\boldsymbol{\omega}}_i \times {}^{i}\boldsymbol{p}_{i+1}\,) - ({}^{i}\boldsymbol{\omega}_i \times {}^{i}\boldsymbol{\omega}_i \times {}^{i}\mathbf{p}_{i+1})\,] \qquad 2.46)$$

- Prismatic : ${}^{i}\mathbf{p}_{i+1}=d_i\hat{x}_i$, ${}^{i}\dot{\boldsymbol{p}}_{i+1} = \dot{d}_i\hat{x}_i$, ${}^{i}\ddot{\boldsymbol{p}}_{i+1} = \ddot{d}_i\hat{x}_i$

$${}^{i+1}\dot{\boldsymbol{v}}_{i+1} = {}_{i+1}R_i\, [\, {}^{i}\dot{\boldsymbol{v}}_i + \ddot{d}_i\hat{x}_i + 2({}^{i}\boldsymbol{\omega}_i \times \dot{d}_i\hat{x}_i\,) + ({}^{i}\dot{\boldsymbol{\omega}}_i \times d_i\hat{x}_i\,) + ({}^{i}\boldsymbol{\omega}_i \quad \times {}^{i}\boldsymbol{\omega}_i \times d_i\hat{x}_i\,)\,] \qquad 2.47)$$

Untuk mencari kecepatan translasi dari pusat massa (center of mass):

$${}^{i+1}\dot{\boldsymbol{v}}_{cm,i+1} = ({}^{i+1}\dot{\boldsymbol{\omega}}_{i+1} \times {}^{i+1}\boldsymbol{p}_{cm}\,) + ({}^{i+1}\boldsymbol{\omega}_{i+1} \times {}^{i+1}\boldsymbol{\omega}_{i+1} \times {}^{i+1}\boldsymbol{p}_{cm}) + {}^{i}\dot{\boldsymbol{v}}_{i+1} \qquad 2.48)$$

Newton/Euler equation untuk gaya dan torsi.

$${}^{i+1}\boldsymbol{F}_{i+1} = m_{i+1}\, {}^{i+1}\dot{\boldsymbol{v}}_{cm,\, i+1} \qquad 2.49)$$

$${}^{i+1}\boldsymbol{N}_{i+1} = \mathbf{M}_{i+1}\,{}^{i+1}\dot{\boldsymbol{\omega}}_{i+1} + ({}^{i+1}\boldsymbol{\omega}_{i+1} \times \mathrm{M}_{i+1}\,{}^{i+1}\boldsymbol{\omega}_{i+1}) \quad 2.50)$$

dengan : **F** =gaya ; **N** = torsi ; **M** = momen inersia

2.3. Motor Serial Servo

Motor serial servo merupakan jenis motor yang membenamkan mikrokontroler didalamnya. Dengan membenamkan mikrokontroler ke dalam rangkaian pengontrolannya maka akan didapatkan banyak fungsi-fungsi tambahan yang membuat motor serial servo ini menjadi lebih unggul jika dibandingkan dengan motor servo konvensional yang pengontrolannya masih mengandalkan sinyal PWM (Pulse Width Modulation). Untuk dapat melakukan pengontrolan motor servo serial, digunakan komunikasi serial. Dengan pemanfaatan komunikasi serial dan fitur mikrokontroler yang dibenamkan ke dalam motor servo serial, didapatkan berbagai keuntungan yakni motor servo serial dapat memberikan berbagai feedback posisi angular actuator motor, torsi, beban dan tegangan yang bekerja, temperature motor. Selain itu penggunaan motor serial servo ini bisa mengurangi beban kerja mikrokontroler karena pengontrolan motor servo serial itu sendiri telah dikonfigurasikan oleh mikrokontroler yang ada di dalam motor serial itu sendiri[2].

2.3.1. Dynamixel RX-24F

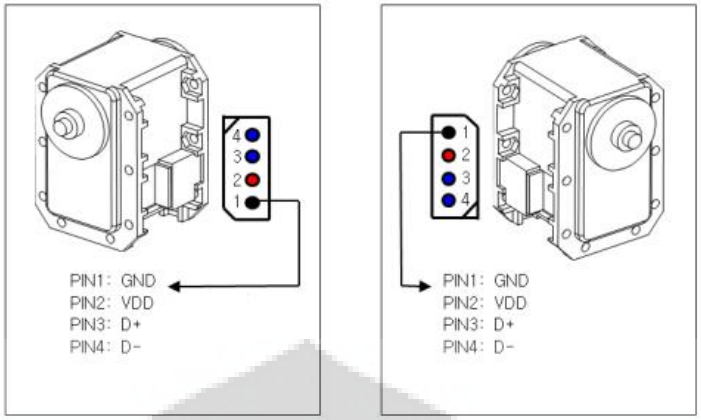
Dalam perancangan tangan robot ini digunakan actuator RX-24F untuk bagian lengan robot dan DC gearmotor untuk bagian jari-jari robot. Pemilihan servo dynamixel RX-24F untuk lengan robot dikarenakan dynamixel RX-24F memiliki torsi yang cukup kuat dan memiliki feedback torsi, temperatur, dan posisi yang dapat dimanfaatkan untuk merasakan tekanan atau sentuhan dari lingkungannya.

Gambar 2.15 Dynamixel RX-24F[14]

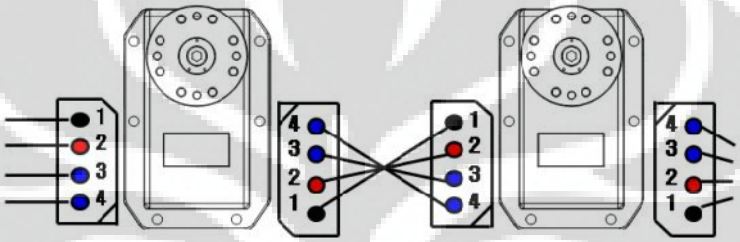
Spesifikasi dynamixel RX-24F adalah sebagai berikut,

- Weight : 67g
- Dimension : 35.5mm x 50.8mm x 41.8mm c. Resolution : 0.29°
- Gear Reduction Ratio : 193 : 1
- Stall Torque : 26kgf.cm (at 12V, 2.4A)
- Running Degree 0° ~ 300°
- Endless Turn
- Running Temperature : -56 ~ +856
- Voltage : 9V~12V
- Command Signal : Digital Packet
- k. Protocol Type : RS485 Asynchronous Serial Communication (8bit,1stop, No Parity) , baut rate default =57600 (double speed mode), ID default = 1.
- Link (Physical) : RS485 Multi Drop Bus
- ID : 254 ID (0~253)
- Communication Speed : 7843bps ~ 1 Mbps
- Feedback : Position, Temperature, Load, Input Voltage, etc.
- Material : Full Metal Gear, Engineering Plastic Body
- Standby current : 50 mA

Konfigurasi pin dynamixel dan wiring link dinamixel dapat dilihat pada gambar berikut.



Gambar 2.16. Pin Assignment Dynamixel RX-24F[14]

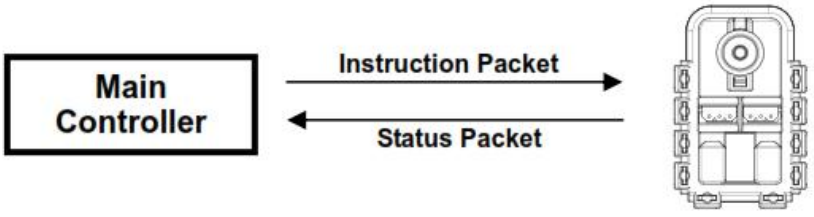


Gambar 2.17. Wire Link Dynamixel RX-24F[14]

User atau pengguna dapat mengontrol dynamixel dengan mengubah data tabel control via paket instruksi. Tabel kontrol berisi data mengenai status dan operasi sekarang. Tabel kontrol dynamixel dapat dilihat dibawah ini.

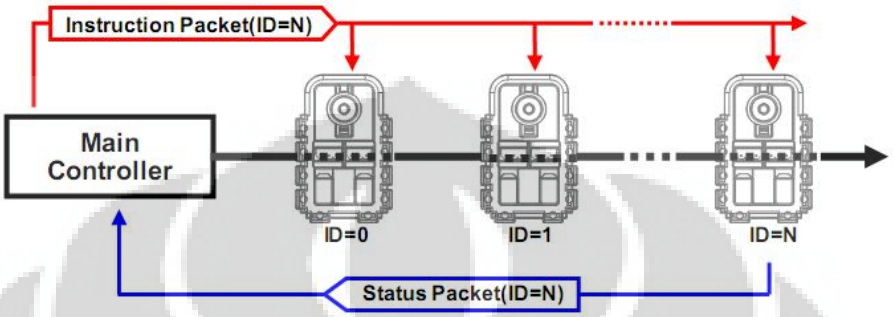
2.3.2. Protokol Serial Komunikasi

Pada sistem komunikasi dynamixel, kontroler utamanya berkomunikasi dengan mengirim dan menerima paket data. Ada 2 macam paket data yang digunakan pada komunikasi dynamixel, yaitu paket instruksi (instruction packet) yang dikirimkan kontroler ke Dynamixel dan paket status (status packet) yang dikirimkan Dynamixel ke kontroler.



Gambar 2.18 Aliran Paket Data dalam Komunikasi dengan Dynamixel[14]

Setiap dynamixel memiliki ID yang berbeda-beda dengan range ID 0-255. Koneksi yang digunakan untuk menghubungkan Dynamixel satu dengan yang lain yaitu dengan menggunakan koneksi daisy chain. Koneksi daisy chain dapat dilihat pada gambar di bawah ini.



Gambar 2.19 Koneksi Multidrop RX-24F[14]

Kontroler berkomunikasi dengan dynamixel dengan menggunakan paket data yaitu

a. Paket Instruksi (*instruction packet*)

Untuk melakukan pengontrolan dynamixel AX-12 diperlukan penyusunan paket-paket instruksi yang sesuai dengan datasheet AX-12. Bentuk umumnya adalah sebagai berikut.

OXFF	0XFF	ID	LENGTH	INSTRUCTION	PARAMETER1	...	PARAMETER N	CHECK SUM

- 0xFF 0xFF

Merupakan indikasi mulainya paket data.

- ID

Merupakan ID dari dynamixel.Tersedia sebanyak 254 ID dengan range dari 0x00 sampai 0xFD.

- Length

Merupakan panjang dari paket dimana nilainya merupakan hasil penjumlahan dari banyak parameter (N) +2.

- Instruction

Instruksi yang diberikan ke dynamixel.Untuk instruksi yang dapat diperintahkan ke dynamixel dapat dilihat pada tabel.

- Parameter0….N

Digunakan jika ada tambahan informasi yang dibutuhkan untuk dikirim pada instruksi itu sendiri.

- Checksum

Perhitungan check sum dilakukan sesuai seperti perumusan di bawah ini

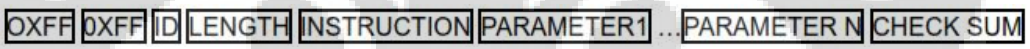
Check Sum = ~ (ID + LENGTH + INSTRUCTION + PARAMETER 1 +… PARAMETER N).

Jika nilai perhitungan lebih dari 255 maka LOW BYTE akan menjadi nilai Check Sum.

*~ menunjukkan negasi misal 1100 = ~0011.

b. Paket Status (Status Packet)

Setelah mengirimkan paket instruksi maka dynamixel akan mengirimkan status paket. Struktur dari status packet tidak berbeda jauh dengan instruction packet seperti terlihat di bawah ini.



- **0xFF 0xFF**

Mengindikasikan mulai pengiriman paket.

- **ID**

ID dari dynamixel yang mengembalikan paket.

- **Length**

Panjang dari paket merupakan banyaknya parameter (N) + 2.

- **Error**

  Jika paket instruksi yang dikirim dapat diterima dengan baik maka bit error akan bernilai 0. Namun jika ada error yang muncul maka bit error yang dikirim balik.

- **Parameter0…N**

  Digunakan bila ada informasi tambahan yang dibutuhkan.

- **Checksum**

Penjelasan sama seperti pada instruction packet.

Terdapat beberapa instruksi yang dapat diberikan kepada dynamixel. Instruksi yang diberikan ke dynamixel berguna untuk berbagai hal. Instruksi-instruksi yang dapat diberikan pada Dynamixel terlihat pada tabel di bawah ini.

Tabel 2.1 Tabel Control RX-24F

Address	Item	Access	Initial Value
0(0X00)	Model Number(L)	RD	12(0X0C)
1(0X01)	Model Number(H)	RD	0(0X00)
2(0X02)	Version of Firmware	RD	?
3(0X03)	ID	RD,W R	1(1X01)
4(0X04)	Baud Rate	RD,W R	1(1X01)
5(0X05)	Return Delay Time	RD,W R	250(0XFA)
6(0X06)	CW Angle Limit(L)	RD,W R	0(0X00)
7(0X07)	CW Angle Limit(H)	RD,W R	0(0X00)
8(0X08)	CCW Angle Limit(L)	RD,W R	255(0XFF)
9(0X09)	CCW Angle Limit(H)	RD,W R	3(0X03)
10(0X0A)	(Reserved)	-	0(0X00)
11(0X0B)	The Highest Limit Temperature	RD,W R	85(0X55)
12(0X0C)	The Lowest Limit Voltage	RD,W R	60(0X3C)
13(0X0D)	The Highest Limit Voltage	RD,W R	190(0XBE)
14(0X0E)	Max Torque(L)	RD,W R	255(0XFF)
15(0X0F)	Max Torque(H)	RD,W R	3(0X03)
16(0X10)	Status Return Level	RD,W R	2(0X02)
17(0X11)	Alarm LED	RD,W R	4(0X04)
18(0X12)	Alarm Shutdown	RD,W R	4(0X04)
19(0X13)	(Reserved)	RD,W R	0(0X00)

20(0X14)	Down Calibration(L)	RD	?
21(0X15)	Down Calibration(H)	RD	?
22(0X16)	Up Calibration(L)	RD	?
23(0X17)	Up Calibration(H)	RD	?
24(0X18)	Torque Enable	RD,W R	0(0X00)
25(0X19)	LED	RD,W R	0(0X00)
26(0X1A)	CW Compliance Margin	RD,W R	0(0X00)
27(0X1B)	CCW Compliance Margin	RD,W R	0(0X00)
28(0X1C)	CW *Compliance slope*	RD,W R	32(0X20)
29(0X1D)	CCW Compliance Margin	RD,W R	32(0X20)
30(0X1E)	Goal Position(L)	RD,W R	[Addr36]value
31(0X1F)	Goal Position(H)	RD,W R	[Addr37]value
32(0X20)	Moving Speed(L)	RD,W R	0
33(0X21)	Moving Speed(H)	RD,W R	0
34(0X22)	Torque Limit(L)	RD,W R	[Addr14]value
35(0X23)	Torque Limit(H)	RD,W R	[Addr15]value
36(0X24)	Present Position(L)	RD	?
37(0X25)	Present Position(H)	RD	?
38(0X26)	Present Speed(L)	RD	?
39(0X27)	Present Speed(H)	RD	?
40(0X28)	Present Load(L)	RD	?
41(0X29)	Present Load(H)	RD	?
42(0X2A)	Present Voltage	RD	?
43(0X2B)	Present Temperature	RD	?
44(0X2C)	Registered Instruction	RD,W R	0(0X00)
45(0X2D)	(Reserved)	-	0(0X00)
46(0X2E)	Moving	RD	0(0X00)
47(0X2F)	Lock	RD,W R	0(0X00)
48(0X30)	Punch(L)	RD,W R	32(0X20)
49(0X31)	Punch(H)	RD,W R	0(0X00)

Keterangan tabel kontrol dynamixel sebagai berikut:

- EEPROM dan RAM

  Data pada area RAM selalu tereset menjadi initial value kapanpun sumber tegangan dinyalakan sedangkan data pada area EEPROM selalu bernilai tetap sesuai nilai yang diset walaupun sumber tegangan dimatikan dan dinyalakan lagi.

- Address

  Address merepresentasikan lokasi data. Untuk membaca atau menulis data pada tabel kontrol, user harus menulis alamat yang benar pada instruksi paketnya.

- Access

  Dynamixel mempunyai dua macam data, yaitu *Read-only data* yang mana sebagian besar digunakan untuk sensing dan *Read-and-Write data* yang digunakan untuk mengendalikan dynamixel.

- Initial Value

  Pada EEPROM area, initial value pada sisi kanan dari tabel kontrol diatas adalah setingan default pabrik. Pada RAM area, initial value pada sisi kanan tabel kontrol adalah sekali pada saat sumber tegangan dinyalakan.

- Highest/Lowest Byte

  Pada tabel kontrol, beberapa data memiliki nama yang sama, tetapi ditambahi dengan (L) or (H) pada akhir tiap nama untuk membedakan alamat (address). Data ini mengharapkan 16 bit, tetapi dibagi menjadi 8 bit tiap-tiap data untuk pengalamatan (low) dan (high). Pengalamatan ini seharusnya ditulis dengan satu paket instruksi dalam satu waktu.

Berikut ini dijelaskan tiap-tiap bagian data yang tersimpan di tiap alamat yang terdapat pada tabel kontrol.

- Alamat 0x00, 0x01

  Menunjukkan nomor model dari dynamixel, untuk kasus ini dynamixel yang digunakan merupakan AX-12, yang memiliki nilai 0x000C (12).

- Alamat 0x02

  Menunjukkan versi Firmware dynamixel.

- Alamat 0x03

  Menunjukkan ID dari Dynamixel. ID ini merupakan ID yang berbeda-beda yang dimiliki Dynamixel. ID yang berbeda ini diperlukan setiap dynamixel yang berada pada jaringan yang sama.

- Alamat 0x04

Baud Rate. Menentukan kecepatan komunikasi. Perhitungannya sesuai dengan rumus di bawah ini.

Speed(BPS) = 2000000/(Address4+1)

Tabel 2.2. Rumus Bautrate

Address4	Hex	Set BPS	Goal BPS	Error
1	0x01	1000000.0	1000000	0.000%
16	0x10	117647.1	115200	-2.124%
34	0x22	57142.9	57600	-0.160%
51	0x33	38461.5	38400	-0.160%

- Alamat 0x05

  Return Delay Time.Waktu yang diperlukan untuk mengirimkan Status Packet setelah Instruction Packet dikirim. Delay timenya dapat dicari dengan menggunakan rumus 2uSec * Address 5 value.

- Alamat 0x06, 0x07, 0x08, 0x09

Operating Angle Limit. Mengkonfigurasikan rentang sudut pada actuator Dynamixel. Posisi tujuan harus berada dalam rentang CW Angle Limit <= Goal Position <= CCW Angle Limit. Error dari Angle Limit akan terjadi jika Posisi akhir di set diluar rentang operasinya.

- Alamat 0x0B

The Highest Limit Temperature.Menunjukkan batas atas dari temperature Dynamixel. Jika temperatur yang terdeteksi melebihi ambang batas maka Overheating Error Bit (bit 2 dari Status packet) akan mengembalikan nili 1 dan Alarm akan diset pada alamat 17, 18. Nilainya dalam derajat Celcius.

- Alamat 0x0C, 0x0D

The Lowest (Highest) Limit Voltage. Merupakan batas bagian bawah dan atas dari rentang tegangan Dynamixel. Jika tegangan yang terdeteksi (Alamat 42) diluar dari batas yang diizinkan, maka Voltage Range Error Bit (Bit 0 dari Status packet) akan mengembalikan nilai 1. dan Alarm akan diset pada alamat 17, 18. Nilai yang ditunjukkan merupakan nilai 10x dari nilai asli, misal nilai 80 menunjukkan nilai tegangan sebenarnya sebesar 8 Volt.

- Alamat 0x0E, 0x0F, 0x22, 0x23

Max Torque. Torsi maksimum dari Dynamixel. Ketika nilainya diset 0, maka Dynamixel akan masuk ke mode Free Run. Ada 2 lokasi dimana torsi maksimumnya didefinisikan, di EEPROM (Alamat 0x0E, 0x0F) dan di RAM (Alamat 0x22, 0x23) . Ketika power dinyalakan maka torsi maksimum yang didefinisikan di EEPROM akan disalin ke lokasi di RAM. Torsi dari Dynamixel dibatasi pada nilai yang diletakkan di RAM (Alamat 0x22, 0x23). Nilai torsi ini juga berpengaruh untuk menentukan kecepatan maksimum dari aktuator pada motor. Motor yang diset dengan nilai torsi besar akan berputar lebih cepat dibandingkan dengan motor yang diset dengan nilai torsi yang lebih kecil. Nilai torsi juga menentukan besaran maksimum beban yang mampu dideteksi oleh perintah read load yang terdapat pada alamat 0x28 dan 0x29. Jika nilai torsi yang diberikan pada motor merupakan ½ dari nilai torsi maksimum, maka beban yang dapat dideteksi maksimum mempunyai besaran sekitar ½ dari beban maksimum.

- Alamat 0x10

Status Return Level. Menentukan apakah Dynamixel akan mengembalikan *Status packet* setelah menerima *Instruction Packet*.

Tabel 2.3. Status Return Level

Alamat 16	Keterangan pengenbalian *status packet*
0	Tidak Merespos semua instruksi
1	Hanya merespon isntruksi READ_DATA
2	Merespon semua instruksi

Pada kasus instruksi yang menggunakan Broadcast ID (0xFE), Status packet tidak akan dikirim balik.

- Alamat 0x11

  Alarm LED. Jika Bit bernilai 1, maka LED akan berkedip jika terjadi error.

- Alamat 0x12

Alarm Shutdown. Jika bit bernilai 1, maka torsi dari dynamixel akan dimatikan ketika terjadi error.

- Alamat 0x14 – 0x17

Callibration. Untuk mengkalibrasi antara potensiometer dengan data yang terdapat pada Dynamixel. Pengguna tidak dapat merubah data ini.

Register-register berikut ini berada pada area RAM.

- Alamat 0x18

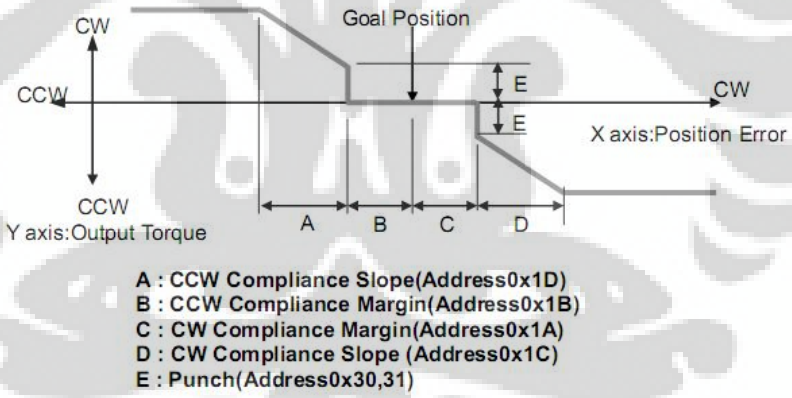
Torque Enable. Ketika power pertama kali dinyalakan, maka Dyamixel akan berada pada mode Torque Free Run (Torsi = 0). Dengan melakukan penyetelan pada alamat 0x18 menjadi 1 maka akan menyalakan torsi.

- Alamat 0x19

LED. Led akan dinyalakan jika diset 1 dan mati jika diset 0.

- Alamat 0x1A – 0x1D

Compliance Margin and Slope. Toleransi Dynamixel didefinisikan dengan menyetel nilai margin dan slope. Fitur ini bisa digunakan untuk menyerap getaran pada shaft output. Gambar di bawah ini menunjukkan bagaimana nilai toleransi dari margin dan slope didefinisikan dengan posisi error dan torsi yang diaplikasikan.



Gambar 2.20 Toleransi Margin dan Slope[14]

Compliance Margin

Mengacu kepada nilai error yang diset antara posisi tujuan dengan posisi actuator dynamixel AX-12 pada saat sekarang. Memiliki nilai range 0-255. Semakin besar nilainya maka perbedaan posisi dynamixel dengan posisi tujuan semakin lebar. Nilai 0-255 merupakan representasi dari nilai toleransi goal yang

dikehendaki. Nilai toleransi goal yang dikehendaki dirumuskan dengan rumus Posisi toleransi = nilai margin x 0,2932 derajat.

Compliance slope

Mengacu kepada arah gerakan dari aktuator Dynamixel AX-12. Nilai pada compliance merepresentasikan level torsi pada posisi tujuan. Nilai dari compliance slope terdiri dari 7 langkah, semakin besar nilainya maka aktuator dari dynamixel akan semakin fleksibel. Nilai compliance slope mempunyai rentang 0-255 dengan pembagian menjadi 7 level torsi seperti yang terlihat pada tabel dibawah ini.

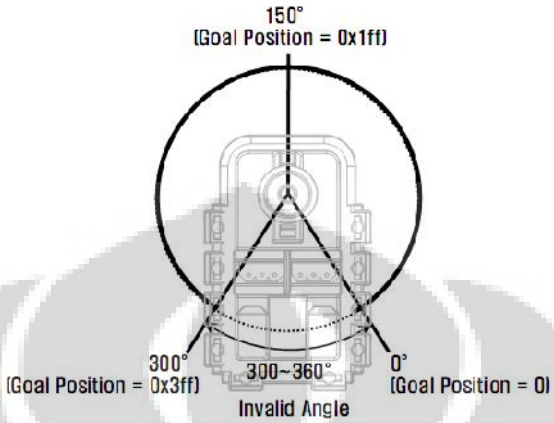
Tabel 2.4. Nilai Slope

Step	Nilai Slope	Nilai Representatif
1	0(0x00)-2(0x02)	2
2	4(0x04)-7(0x07)	4
3	8(0x08)-15(0x0f)	8
4	16(0x10)-31(0x1f)	16
5	32(0x20)-63(0x3f)	32
6	64(0x40)-127(0x7f)	64
7	128(0x80)-254(0xfe)	128

Dengan merubah-rubah nilai tersebut maka kelenturan dari actuator dynamixel untuk menyerap getaran dapat divariasikan. Ketujuh level pada compliance slope akan mempengaruhi besar torsi dari dynamixel ketika aktuatornya akan mencapai posisi tujuan yang diperintahkan kepadanya. Compliance slope akan membagi torsi menjadi 1-7 level. Dengan pembagian level tersebut akan mempengaruhi kecepatan actuator dynamixel ketika menuju posisi tujuan dan perbedaan posisi actuator dari posisi tujuan sebenarnya. Selain itu dengan mengubah-ubah nilai compliance slope, berarti mempengaruhi distribusi torsi disekitar goal position. Seperti yang telah dijelaskan sebelumnya, nilai slope yang kecil menunjukkan aktuator dynamixel lebih mudah digerakkan dibandingkan dengan nilai slope yang kecil.

- Alamat 0x1E, 0x1F

Goal Position. Meminta posisi dari Dynamixel untuk bergerak sesuai dengan yang diperintahkan. Menyetel nilai ini menjadi 0x3FF akan menggerakkan aktuator Dynamixel pada posisi 300 derajat.



Gambar 2.21. Goal Position Dynamixel RX-24F[14]

- Alamat 0x20, 0x21

Moving Speed. Menyetel kecepatan dari pergerakan Dynamixel untuk mencapai posisi yang diperintahkan.Jika menyetel pada nilai 0x3FF maka kecepatannya mencapai 114 RPM.

- Alamat 0x24, 0x25

Present Position. Menunjukkan posisi sekarang pada actuator Dynamixel.

- Alamat 0x26, 0x27

Present Speed. Menunjukkan kecepatan aktual dari actuator Dynamixel.

- Alamat 0x28, 0x29

Present Load. Menunjukan magnitude dari beban yang beroperasi pada Dynamixel. Bit ke 10 menunjukkan arah dari bebannya. Beban yang dideteksi dipengaruhi oleh nilai torsi yang diatur oleh pengguna yaitu besaran yang terdapat pada alamat 0x22 dan 0x23. Nilai torsi ini menjadi nilai acuan bagi dynamixel untuk dapat menahan beban yang bekerja pada aktuatornya. Jika nilai torsi di set setengah dari nilai maksimum yakni sekitar 8 kgf.cm maka ketika aktuator pada motor diberi beban lebih besar dari 8 kgf.cm atau setara dengan nilai 512 (0x1ff) maka dynamixel tidak akan dapat bergerak dan beban yang terdeteksi juga merupakan nilai torsi yang diset yakni 512. Jadi nilai beban maksimal yang dapat dideteksi oleh dynamixel dipengaruhi oleh nilai torsi.

Tabel 2.5 Tabel Beban[14]

BIT	15-11	10	9	8	7	6	5	4	3	2	1	0
Nilai	0	arah load	nilai beban (load)									

- Alamat 0x2A

Present Voltage. Menunjukkan tegangan yang diaplikasikan kepada motor pada saat ini. Nilainya merupakan 10x dari nilai asli.Misal 10 V menjadi 0x64 (100).

- Alamat 0x2B

Present Temperature. Menunjukan temperature internal dari actuator Dynamixel dalam derajat Celcius.

- Alamat 0x2C

Registered Instruction. Set menjadi 1 ketika instruksi ditugaskan oleh perintah REG_WRITE. Set menjadi 0 ketika perintah instruksinya telah selesai oleh Action command.

- Alamat 0x2E

Moving. Set menjadi 1 ketika actuator Dynamixel bergerak dengan powernya sendiri.

- Alamat 0x2F

Lock. Jika diset 1, maka hanya alamat 0x18 sampai 0x23 yang bisa dituliskan, sedangkan area lain tidak dapat ditulis. Jika di kunci, maka cara supaya tidak terkunci adalah dengan mematikan power Dynamixel-nya.

- Alamat 0x30, 0x31

Punch. Menunjukkan nilai arus minimum yang disuplai ke motor selama beroperasi. Nilai inisialisasi diset 0x20 dan nilai maksimum 0x3FF.

Instruksi yang dapat dituliskan ke dalam paket instruksi dapat dilihat pada tabel di bawah ini.

Tabel 2.6. Instruksi Paket[14]

Instruksi	Fungsi	Value
PING	Digunakan untuk mendapatkan status paket	0x01
READ DATA	Membaca nilai untuk kontrol table	0x01
WRITE DATA	Menuliskan nilai ke kontrol table	0x03

REG WRITE	Sama seperti WRITE DATA, tetapi menunggu instruksi ACTION	0x04
ACTION	Menjalankan aksi yang di instruksikan REG WRITE	0x05
RESET	Mengatur kembali setelan motor ke kondisi *default*	0x06
STNC WRITE	Mengontrol banyak motor sekaligus	0x07

Dalam skripsi ini, instruksi yang sering digunakan adalah instruksi untuk membaca posisi aktuator motor dan membaca beban yang terdeteksi. Berikut ini adalah contoh instruksi yang digunakan dalam skripsi ini

Menggerakkan servo dengan ID 01 ke posisi 207,9 derajat

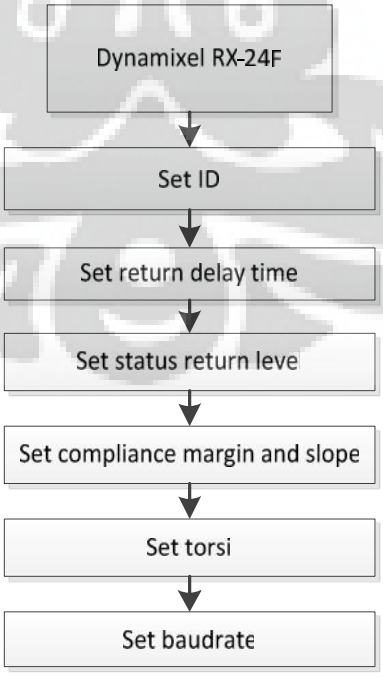
Goal position address 0x1E

Data 0x252

Paket instruksi FF FF 01 05 03 1E 52 02 84

Paket status FF FF 01 02 00 FC

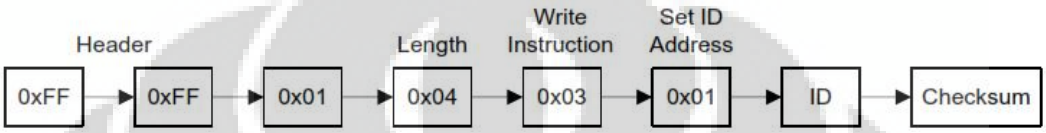
Dalam perancangan compliance control, actuator dynamixel ini diset dengan parameter-parameter yang ditentukan.



Gambar 2.22. Algoritma Konfigurasi Awal Dynamixel RX-24F

a. Set ID

Nilai ID yang di set pada tiap dynamixel harus berbeda karena ID ini yang akan menjadi identitas tiap dynamixel agar dapat di instruksi dengan perintah yang berbeda-beda sesuai dengan pengalamatannya. Jumlah total dynamixel pada lengan robot ada 6 buah yang masing masing lengan terdiri dari 3 dynamixel. Untuk lengan kanan ID nya di set nilai 2, 4, dan 6. Sedangkan untuk lengan kiri ID di set 1,3, dan 5. Berikut urutan instruksi untuk mengeset ID dynamixel.



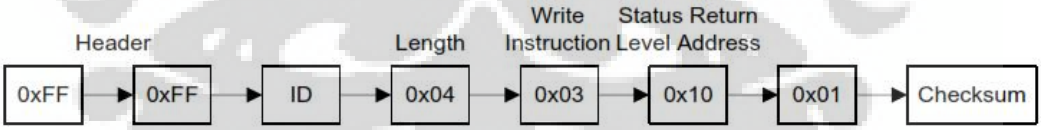
Gambar 2.23. Instruksi paket set ID

b. Set Return Delay Time

Return delay time diset dengan nilai terkecil yaitu 2 us untuk menghemat waktu proses looping.

c. Set Status Return Level

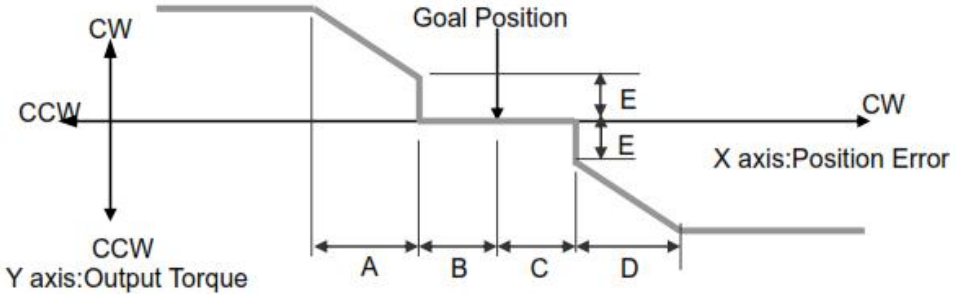
Status Return Level di set 1 yaitu hanya mengembalikan status packet untuk perintah membaca saja (READ_INSTRUCTION) agar lebih efisien. Berikut urutan instruksinya.



Gambar 2.24. Instruksi paket set Status Return Level

d. Set Compliance Margin And Slope

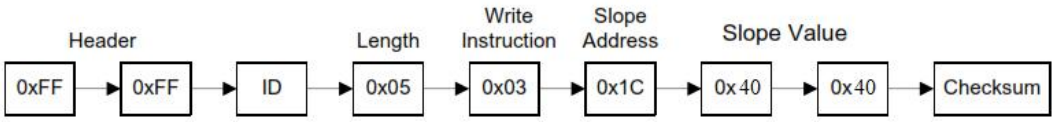
Konfigurasi nilai compliance margin dan slope yang digunakan dapat dilihat pada gambar dibawah ini.



Gambar 2.25. Konfigurasi compliance margin dan slope[14]

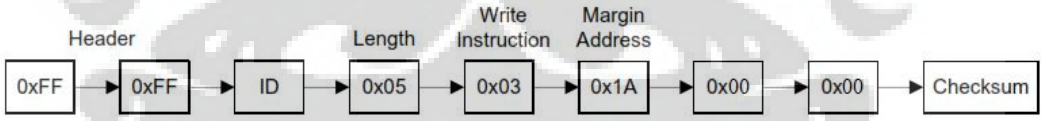
A dan D merupakan CCW dan CW *Compliance slope*. Nilai slope menunjukkan level torsi pada saat posisi aktuator dari RX-24F mendekati posisi tujuannya. Seperti yang terlihat pada gambar 3.4, nilai *compliance slope* akan mengatur besarnya torsi pada saat aktuator dari dynamixel sesaat sebelum mencapai posisi tujuan. Pada bagian dasar teori, secara default, motor dynamixel membagi *compliance slope* menjadi 7 level yang berbeda. Semakin besar nilainya maka akan menyerap getaran tetapi dengan konsekuensi error yang dihasilkan akan semakin besar. Karena ketika nilai *compliance slope* diset dengan nilai maksimal yaitu 254 (0xfe) maka tingkat kemiringan yang diperlihatkan pada bagian A dan D akan semakin landai. Sehingga mempengaruhi daya dorong pergerakan dari aktuator dynamixel untuk mencapai posisi tujuannya, namun dengan menyetel nilai torsi yang lebih landai maka aktuator dari dynamixel mampu menyerap getaran lebih baik. Berlawanan dari hal tersebut, dengan nilai *compliance slope* yang lebih kecil maka kemiringan pada bagian A dan D akan lebih curam. Ini akan menjaga daya dorong dynamixel untuk mencapai posisi tujuan lebih baik namun akan membuat dynamixel lebih rentan terhadap getaran dari efek daya dorongnya. Konfigurasi yang digunakan dalam perancangan sistem bilateral menggunakan RX-24F adalah menyetel nilai slope-nya menjadi 64 (0x40). Nilai tersebut menjaga putaran motor agar selalu memiliki nilai torsi yang cukup untuk mencapai posisi tujuan. Apabila nilai torsi tidak cukup besar dalam mencapai posisi tujuan maka aktuator akan berhenti sebelum mencapai posisinya dan ini akan menyebabkan nilai error yang lebih besar. Pengaturan besarnya nilai *compliance slope* dilakukan dengan menuliskan besaran yang diinginkan pada alamat 0x1D dan 0x1C pada dynamixelnya. Berikut ini adalah paket instruksi

yang dikirimkan ke RX-24F untuk mengkonfigurasikannya sesuai yang telah dijelaskan sebelumnya.



Gambar 2.26. Instruksi Paket Set Nilai Slope

Kemudian untuk B dan C merujuk kepada CCW dan CW Compliance margin. Dengan memberikan nilai compliance margin maka toleransi error yang diberikan kepada aktuator dynamixel dapat dilakukan. Nilai compliance margin dapat diset dengan rentang (0 – 254). Nilai tersebut merepresentasikan nilai error yang diizinkan dengan perumusan. Toleransi Error = nilai compliance margin x0.2932 derajat Nilai tersebut kemudian menjadi acuan bagi dynamixel untuk tidak memberikan gaya dorong ketika aktuatornya mencapai nilai toleransi error. Misal nilai toleransi error yang diset adalah 1 derajat untuk CW maupun CCW. Ketika dynamixel diperintahkan mencapai posisi 180 derajat ketika aktuatornya mencapai posisi 179 de rajat maka daya dorongnya akan dihentikan karena nilai toleransi error yang telah diset sebelumnya. Besarnya nilai yang digunakan untuk compliance margin adalah sebesar 0 derajat untuk masing-masing CCW dan CW sehingga toleransi errornya nol. Artinya tidak ada error. Berikut ini urutan instruksi yang diberikan.



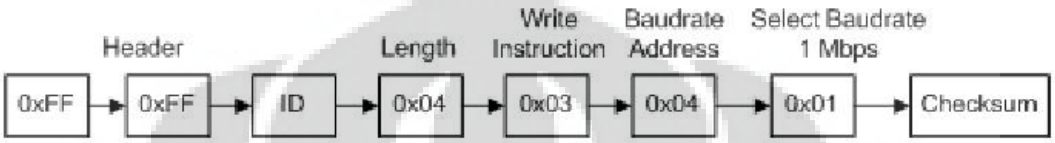
Gambar 2.27. Instruksi Paket Set Nilai Margin

e. Set Torsi

Kondisi *default* nilai torsi yang tersimpan di EEPROM adalah 1024(0x3FF). Sumber tegangan yang diberikan kepada dynamixel RX-24F adalah 12 Volt. Dengan tegangan 12 volt, torsi yang diberikan dynamixel RX-24F adalah 26 kgf.cm. Nilai torsi ini kemudian dinormalisasikan dan akan menjadi feedback pada compliance control sebagai torsi disturbance. Nilai torsi di set dengan nilai maksimal 1024.

f. Set Bautrate

Baudrate yang digunakan adalah 1 Mbps. Dengan baudrate sebesar ini komunikasi yang dilakukan dapat menghemat waktu sampling. Dengan komunikasi sebesar 1 Mbps, mampu dilakukan transfer byte sebesar 125 byte /ms. Sehingga bisa meningkatkan performa dari sistem compliance control. Untuk mengeset nilai baudrate menjadi 1 Mbps dapat dilakukan dengan mengirimkan instruksi paket seperti di bawah ini ke dynamixel.



Gambar 2.28. Instruksi Paket Set Baudrate

2.4. Motor DC gearbox 6Volt

Motor yang digunakan sebagai actuator jari-jari tangan robot adalah motor DC gearbox 6 Volt. Menggunakan motor tersebut karena selain ukurannya kecil dan ringan, torsinya pun cukup kuat untuk menarik pegas yang ada pada jari-jari robot. Dengan menggunakan rangkaian tanbahan berupa current sensor untuk motor, maka torsi motor dapat didapat dari rangkaian curret sensor tersebut.

Gambar 2.29. DC Gearmotor

Dimensi:

- Ukuran : 24 x 10 x 12 mm
- Berat : 0.34 oz
- diameter Shaft : 3 mm

Spesifikasi Umum :

- Gear ratio : 5:1
- Free-run speed (6V) : 6000 rpm
- Free-run current (6V) : 120 mA
- Stall current (6V) : 1600 mA

- Stall torque (6V) : 2 oz•in

2.5. Mikrokontroler Atmega 128

Mikrokontroler yang digunakan adalah seri ATmega128 karena ATmega128 memiliki fitur 2 buah komunikasi serial yang digunakan untuk komunikasi ke dynamixel dan ke master.

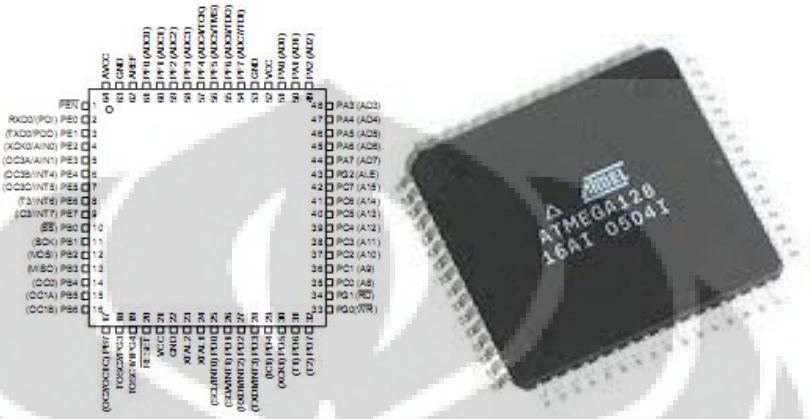
Sebuah mikrokontroler merupakan single chip mikoprosessor atau mikrokomputer dalam sebuah Integrated Circuit (IC) yang terdiri dari CPU sederhana dikombinasikan dengan fitur-fitur pendukung seperti Kristal osilator, pewaktuan, watchdog timer, serial dan analog I/O, dan fitur-fitur pendukung lain. Sebuah mikrokontroler juga dapat dikatakan sebagai sistem yang berdiri sendiri dengan sebuah prosesor, memori, serta peripheral yang dapat digunakan dengan sebuah embeded system (dengan dilengkapi perangkat lunak).

Mikrokontroler yang dipakai dalam perancangan tangan robot ini adalah seri ATmega128. ATmega128 merupakan salah satu seri mikrokontroler keluarga AVR keluaran Atmel yang paling banyak digunakan. Hal ini dikarena selain harganya terjangkau, mikrokontroler ini juga memiliki fasilitas onchip memory. ATmega128 adalah mikrokontroler CMOS 8 bit berdaya rendah, berbasiskan arsitektur RISC dari AVR yang telah ditingkatkan. Dengan menjalankan instruksi dalam satu clock cycle, ATmega16 dapat mendekati 1 MIPS per MHz[15].

Inti dari AVR mengkombinasikan kumpulan instruksi dengan 32 general purpose working registers (GPR). Semua register secara langsung terkoneksi dengan Arithmetic Logic Unit (ALU), memungkinkan akses dua register bebas dalam sekali instruksi yang dieksekusi dalam satu clock cycle. Hasilnya adalah efisiensi hamper sepuluh kali lipat dari mikrokontroler konvensional CISC[15].

ATmega128 mempunyai fitur sebagai berikut: 128K bytes In-System Programmable Flash Program memory dengan kemampuan Read-While-Write, 4Kbytes EEPROM, 4K byte SRAM, 32 general purpose I/O, 32 GPR, sebuah JTAG interface, mendukung On-chip Debugging and programming, delapan buah Timer/Counters yang fleksibel dengan mode pembanding, Internal dan Eksternal Interupsi, dua buah USART, sebuah byte berorientasi Two-wire Serial Interface, dan 10-bit ADC 8-channel dengan pilihan tingkatan input dengan gain yang dapat

diatur, sebuah programmable Watchdog Timer dengan Internal Oscillator, sebuah SPI serial port, dan 6 software mode selectable power saving. Mode Idle menghentikan CPU saat penggunaan USART, Two-wire interface, A/D Converter, SRAM, timer/Counters, SPI port, dan system interupsi untuk melanjutkan proses[15].



Gambar 2.30. ATmega 128[15]

Komunikasi yang digunakan dan didukung oleh RX-24 adalah komunikasi asynchronous. Komunikasi asynchronous memerlukan start bit dan end bit yang ditambahkan ke dalam data bit agar receiver dapat menentukan waktu dan timing dari tiap bitnya. Gambar di bawah ini menjelaskan salah satu contoh dari komunikasi asynchronous.

State	Idle	Idle	Active	Active	Active	Active	Active	Active	Active	Active	Active	Active	Idle
Logic Level	1	1	0	1	0	0	0	0	0	1	0	1	1
Bit Type			Start	Data	Data	Data	Data	Data	Data	Data	Data	Stop	
Data Bit Number				0	1	2	3	4	5	6	7		

Gambar 2.31. Komunikasi Asynchronous

Terlihat pada grafik di atas bahwa pada kondisi diam, serial komunikasi bernilai 1 dan bernilai 0 untuk menunjukkan komunikasi serial akan dimulai.

Bitpertama bernilai 1 dan kemudian akan diikuti 8 bit berikutnya. Nilai stop bit bernilai 1 sama seperti keadaan ketika sedang diam.

Fitur-fitur Atmega128 yang digunakan di skripsi ini adalah

1. UART1 (Universal Asynchronous Receiver Transmitter 1)

UART1 difungsikan sebagai sarana komunikasi antara mikrokontroler dengan Dynamixel RX-24. Komunikasinya mempunyai baudrate sebesar 1 Mbps. Register yang digunakan pada UART1 adalah UCSR1A, UCSR1B, UBBR1, dan UDR1. Penjelasan dari tiap-tiap bagiannya akan dijelaskan kemudian.

2. UART0 (Universal Asynchronous Receiver Transmitter 0)

UART0 difungsikan sebagai sarana komunikasi antara mikrokontroler tangan robot dengan mikrokontroller master (mikrokontroller utama). Komunikasi yang digunakan sebesar 38.400 bps. Register yang digunakan pada UART0 adalah UCSR0A, UCSR0B, UBBR0 dan UDR0.

Pada mikrokontroler berbasis AVR, UART-nya mempunyai 2 register. Berikut ini adalah penjelasan register-register yang digunakan dalam perancangan compliance control dalam skripsi ini.

• UART Control Register (UCSRB dan UCSRA)

UCSRB digunakan untuk mengatur fungsi dari UART. Terdiri dari 7 bit untuk mengatur tiap-tiap fungsi dan inisialisasi yang dilakukan dalam komunikasi asynchronous. Bit-bit tersebut tersusun dalam grafik di bawah ini

Tabel 2.7 Register UCSRB

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
RXCIE	TXCIE	UDRIE	RXEN	TXEN	UCSZ2	RXB8	TXB8

Tabel 2.8 Deskripsi Register UCSRB

RXCIE	Jika nilainya diset menjadi 1, maka setiap selesai menerima data akan terjadi interrupt
TXCIE	Jika nilainya diset menjadi 1, maka setiap selesai mengirim data akan terjadi interrupt
UDRIE	Setiap register UDRE kembali bernilai 0 maka akan terjadi interrupt
RXEN	Diset 1 untuk mengaktifkan pin Rx
TXEN	Diset 1 untuk mengaktifkan pin Tx
UCSZ2	Digunakan untuk mengeset banyak data dalam bit
RXB8	Diaktifkan untuk menerima data dalam mode 9 bit
TXB8	Diaktifkan untuk mengirim data dalam mode 9 bit

UCSRA merepresentasikan kondisi UART pada kondisi aktual. Dengan melihat kepada bit-bit pada register UCSRA, dapat diketahui apakah UART sedang menerima, mengirim data atau siap untuk mengirim data. Berikut ini adalah register-register yang berada dalam UCSRA.

Tabel 2.9 Register UCSRA

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
RXC	TXC	UDRE	FE	DOR	PE	U2X	MPCM

Tabel 2.10 Deskripsi Register UCSRA

RXC	Jika bernilai 1 maka telah selesai menerima data
TXC	Jika bernilai 1 maka telah selesai mengirim data
UDRE	Jika bernilai 1 maka register UDR sudah kosong
FE	Jika bernilai 1 maka ada framing error
DOR	Jika bernilai 1 maka ada error
PE	Parity error
U2X	Diset 1 untuk melipatgandakan kecepatan baudrate
MPCM	Diset untuk mengijinkan multi proses

- UART Baud Rate Register (UBBRH dan UBBRL)

Register ini digunakan untuk menentukan baudrate yang digunakan selama melakukan proses pengiriman dan penerimaan data. Untuk menghitung baudrate

yang ingin digunakan dalam komunikasi serial, digunakan rumus matematis seperti di bawah ini

UBBR = (System Clock / (16 x Baudrate)) – 1

3. Timer / Counter

Fitur Timer / counter yang terbenam dalam chip mikrokontroler digunakan dalam penentuan waktu pencuplikan serta. Pada skripsi ini digunakan 2 timer yang masing-masing akan dijelaskan sebagai berikut

- Timer0

Timer 8 bit yang biasa difungsikan untuk melakukan pencuplikan dalam sebuah program. Timer0 digunakan untuk sampling time dari komputasi compliance control.

- Timer1

Merupakan timer 16 bit. Dalam skripsi ini Timer1 difungsikan sebagai penentu waktu cuplik (sampling time) dalam proses looping sistem compliance control pada lengan robot . Pencuplikan dilakukan setiap 20 ms.

Untuk mengeset nilai timer1 agar bisa memerintahkan mikrokontroler untuk mengambil data dan melakukan perhitungan sistem compliance control digunakan rumus sebagai berikut.

TCNT1=(1+0xFF) – (waktu * (XTAL/PRESCALER))

Variable “waktu” adalah waktu interrupt yang di inginkan.

Nilai clock mikrokontroler yang digunakan dalam perancangan adalah

Nilai clock = Clock Sistem / Prescalar

Nilai clock = 16 MHz / 1024

Nilai clock = 15.625 Hz

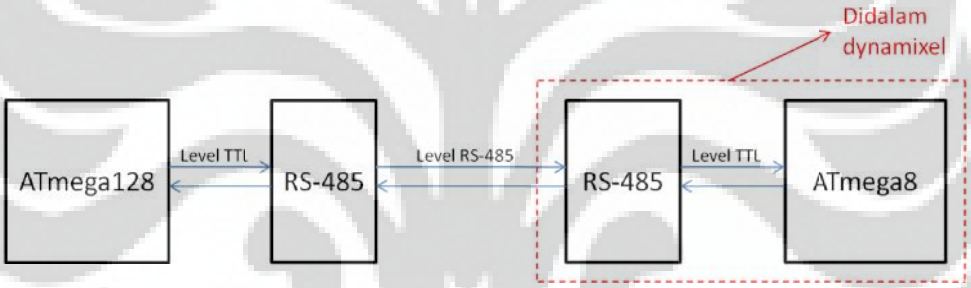
Untuk mendapatkan periode sebesar 20 ms maka digunakan counter 2 kali dari sampling time timer 10 ms. Berikut didapat nilai TCNT1 untuk interrupt timer 10ms.

TCNT1 = 256 – (10 ms*15.625 Hz)

TCNT1 = 0x64

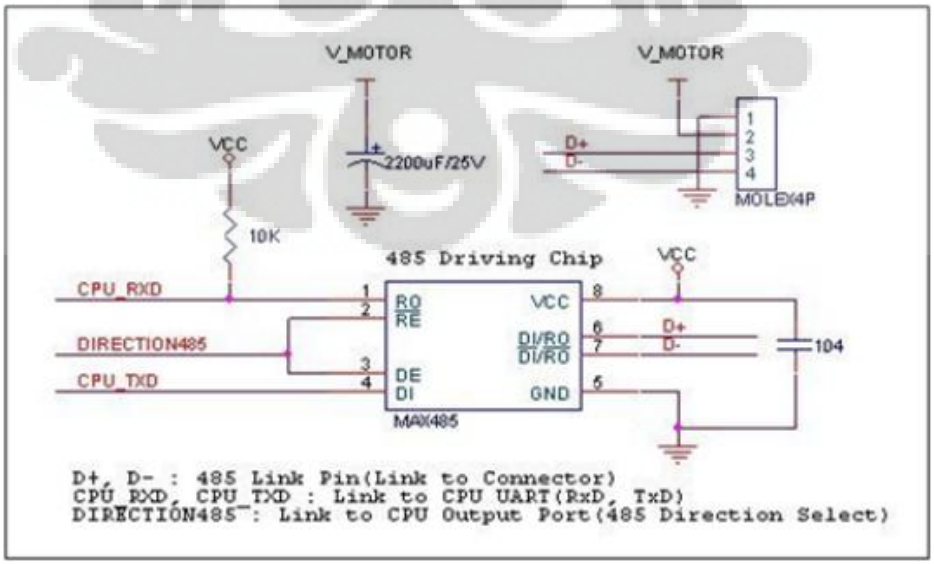
2.6. Perangkat Komunikasi (RS-485)

Perangkat komunikasi yang digunakan untuk menghubungkan antara mikrokontroler dengan dynamixel adalah rangkaian RS-485. RS-485 berfungi untuk merubah level tegangan TTL (low 0 volt dan high 5 volt) ke level tegangan RS-485 (low -5 volt dan high +5volt). Prinsip komunikasinya adalah paket data (data serial) yang dikirim melalui mikrokontroller (level TTL) akan dikonversi ke dalam level RS-485 melalui rangkaian RS-485 yang dibuat kemudian data akan ditransimisikan ke rangkaian RS-485 yang ada didalam dynamixel (RS-485 internal) baru kemudian dikonversi lagi ke level TTL menuju mikrokontroller ATmega8 yang ada didalam dynamixel. Didalam dynamixel itu sendiri sebenarnya sudah ada rangkaian controller sendiri berupa rangkaian ATmega8 dan rangkaian interface RS-485. Jadi kalau kita gambarkan diagram alir dari komunikasi antara mikrokontroler dengan dinamixel adalah sebagai berikut :



Gambar 2.32. Diagram Alir Komunikasi

Koneksi USART RS-485 sebagai berikut:



Gambar 2.33. Rangkaian RS-485[14]

Power Dynamixel disuplai via Pin1(-), Pin2(+). Dari sirkuit diagram diatas, arah data sinyal TxD dan RxD dalam level TTL ditentukan berdasarkan level direction 485 sebagai berikut:

- DIRECTION485 Level = High: signal TxD adalah output ke D+ dan D-.
- DIRECTION485 Level = Low: signal D+ and D- adalah output ke RxD.
- VCC = 5 Volt, V_MOTOR sama dengan tegangan input dynamixel (12Volt)

BAB III
PERANCANGAN

Tangan robot yang digunakan pada skripsi ini memiliki 3 joint. Aktuator yang digunakan berupa duynamixel RX-24F dan motor DC gearbox 6 V. Controller yang digunakan berupa mikrokontroller ATmega 128. Untuk komunikasi serial antara mikrokontroller dan Dynamixel digunakan interface RS-485. Pada bab ini akan dibahas mengenai konfigurasi perangkat keras yang digunakan dan algoritma compliance control.

3.1. Perancangan Perangkat Keras

Pada subbab ini, akan dijelaskan tentang sistem perangkat keras secara kesuluruhan/kompleks meliputi sistem mekanik dan elektrik robot.

3.1.1. Sistem mekanik Robot

- **Lengan robot**

Untuk lengan robotnya digunakan 3 buah dynamixel RX-24F sebagai aktuatornya. Berarti lengan robot memiliki 3 derajat kebebasan untuk bergerak. Lengan robot dapat bergerak kedepan, keatas, dan kesamping. Berikut gambar Aslinya.

Gambar 3.1. Lengan Robot

Spesifikasi lengan robot :

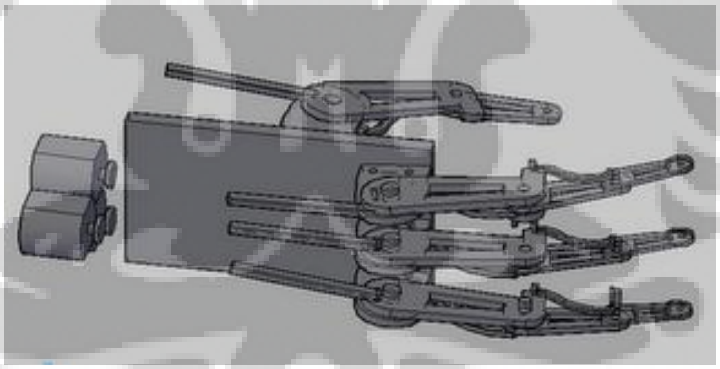
- Panjang Lengan 1 (L1) = 18.5 cm
- Panjang Lengan 2 (L2) = 18.5 cm

- Berat Langan 1 (L1) = 176 gram
- Berat Lengan 2 (L2) = 190 gram

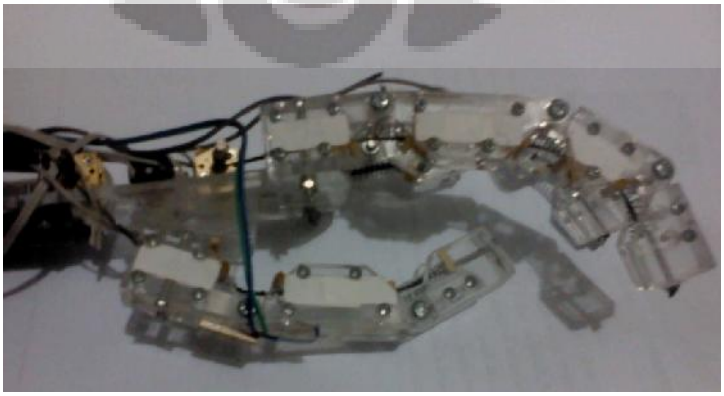
Pada lengan bagian atas terdiri dari 2 buah servo dengan arah rotasi yang berbeda, 1 servo untuk arah depan/belakang dan 1 servo untuk arah kesamping sehingga memungkinkan tangan robot untuk bergerak dalam ruang tiga dimensi yaitu sumbu x, y, dan z. Kemudian untuk lengan bagian bawah hanya ada 1 buah servo yang memungkinkan gerakan rotasi kedepan/kebelakang. Jadi total jumlah servo yang dipakai untuk satu buah tangan robot adalah 3 buah.

- **Telapak tangan robot (gripper)**

Telapak tangan didesain agar robot dapat menggenggam benda. Telapak tangan robot terdiri dari telapak dan jari-jari robot yang jumlahnya 3 buah. Aktuator yangdigunakan untuk menggerakan jari-jari tangan adalah DC gearmotor. Tiap-tiap jari digunakan satu buah motor DC dan totalnya ada 3 buah. Dengan tambahan trangkaian current sensor, maka besarnya torsi dari gripper dapat diperoleh dan compliance control dapat diterapkan. Berikut gambar designnya serta gambar aslinya.



Gambar 3.2. Design Telapak Tangan Tobot

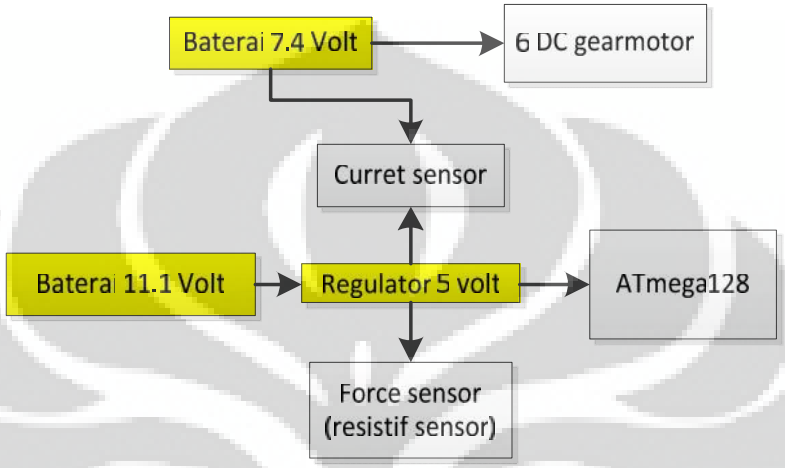


Gambar 3.3. Gambar Asli Telapak Tangan Robot

3.1.2. Sistem Elektrik Robot

Sistem elektrik dari compliance control yang dirancang ini digambarkan dalam blok diagram power distribution dan blok diagram elektrik.

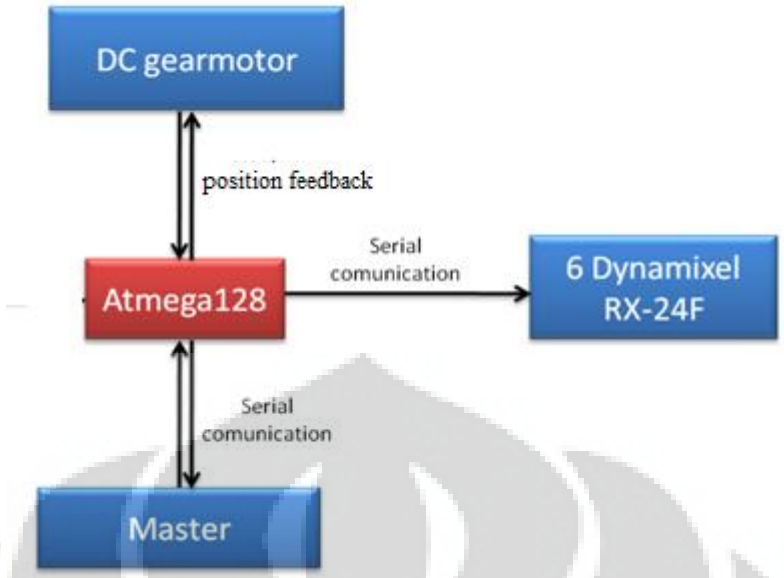
Berikut gambar diagram power distribution tangan robot.



Gambar 3.4. Power Distribution Diagram

Baterai yang digunakan adalah baterai jenis Li-Po 11.1 Volt. Baterai 11.1 volt langsung digunakan untuk menyuplai dynamixel tanpa diregulasi terlebih dahulu sedangkan untuk ke mikrokontroler dan sensor lainnya harus diregulasi terlebih dahulu menggunakan regulator 5 Volt. Regulator yang digunakan adalah IC LM7805.

Diagram Elektrik Sistem Tangan Robot sebagai berikut.



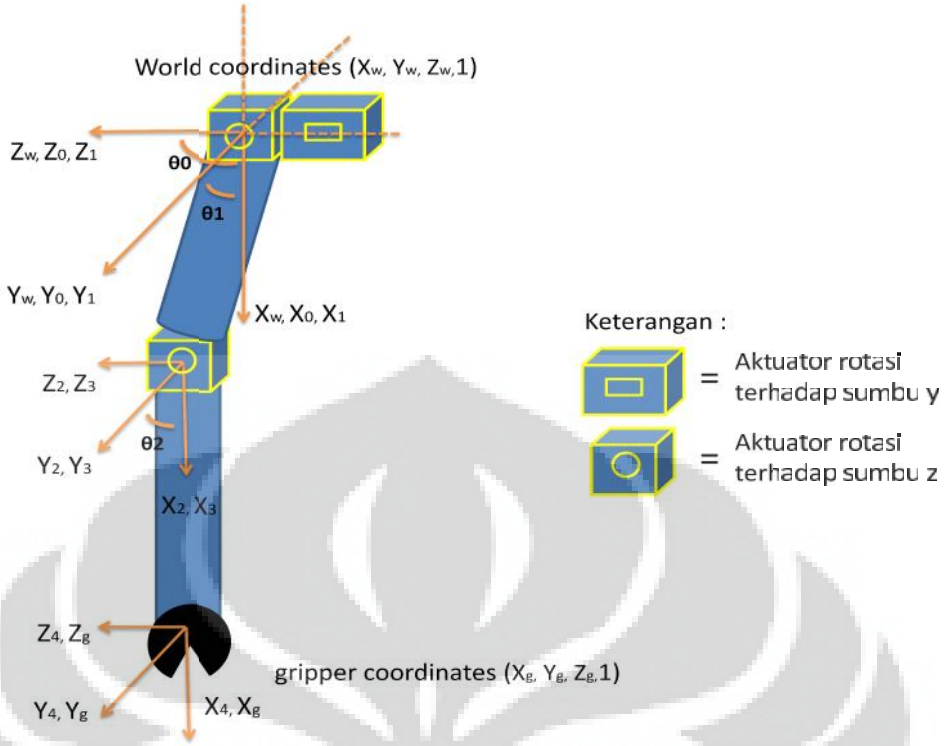
Gambar 3.5. Diagram Elektrik Tangan Robot

Pada diagram elektrik diatas, semua sensor yang digunakan merupakan sensor torque/force. Nilai torsi/gaya bisa didapatkan dari konversi nilai arus, resitansi maupun nilai serial dari sensor. Mikrokontroller yang digunakan adalah jenis AVR ATmega-128 yang memiliki 2 buah fitur USART, yaitu USART0 dan USART1. USART1 digunakan untuk komunikasi dengan dynamixel sedangkan USART0 digunakan untuk komunikasi dengan master.

3.2. Forward Kinematik Menggunakan Homogeneous Transformation

Persamaan forward kinematik tangan robot dapat dicari dengan menggunakan metode Homogeneous Transformation. Homogeneous Transformation merupakan suatu metode untuk mendapatkan titik koordinat tertentu dari suatu koordinat frame yang dipandang dari koordinat frame yang lain atau disebut juga *relative motion*.

Dengan menggunakan metode Homogeneous Transformation, maka dapat dilakukan konversi antara *gripper coordinates* dan *world coordinates* sebagai fungsi sudut dari seluruh joint.



Gambar 3.6. Ilustrasi Koordinat Frames Lengan Robot

Sekarang dilakukan langkah-langkah perhitungannya. Untuk mentransformasikan frame dari *world coordinates* ke *gripper coordinates,* langkah pertama yang dilakukan adalah merotasi frame $(X_w,Y_w,Z_w,1)$ ke frame $(X_0,Y_0,Z_0,1)$ terhadap sumbu z, kemudian merotasi terhadap sumbu y (anggap joint 1 dan 2 dalam 1 titik koordinat sehingga tidak ada translasi frame), kemudian translasi dengan jarak L_1 (17cm) sepanjang sumbu x, kemudian rotasi terhadap sumbu z, dan terakhir translasi dengan jarak L_2 (25cm) sepanjang sumbu x. Sebut langkah tersebut sebagai *relative motion* sehingga cara penyelesaianya adalah kalikan seluruh matriks dari kiri ke kanan secara berurutan. Maka kita dapat persamaan matriks transformasinya sebagai berikut:

$$T = T_w^0 . T_0^1 . T_1^2 . T_2^3 . T_3^4 \qquad 3.8)$$

Dimana :

- ✓ T_w^0 adalah rotasi frame (X_w,Y_w,Z_w,1) ke frame (X_0,Y_0,Z_0,1) terhadap sumbu z.
- ✓ T_0^1 adalah rotasi frame (X_0,Y_0,Z_0,1) ke frame (X_1,Y_1,Z_1,1) terhadap sumbu y.
- ✓ T_1^2 adalah translasi frame (X_1,Y_1,Z_1,1) ke frame (X_2,Y_2,Z_2,1) dengan jarak L_1 sepanjang sumbu x.
- ✓ T_2^3 adalah rotasi frame (X_2,Y_2,Z_2,1) ke frame (X_3,Y_3,Z_3,1) terhadap sumbu z.
- ✓ T_3^4 adalah translasi frame (X_3,Y_3,Z_3,1) ke frame (X_4,Y_4,Z_4,1) dengan jarak L_2 sepanjang sumbu x.

Maka maka didapat persamaan matriks transformasi sebagai berikut.

$$T = \begin{bmatrix} cos\theta_1 & -sin\theta_1 & 0 & 0 \\ sin\theta_1 & cos\theta_1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} cos\theta_0 & 0 & sin\theta_0 & 0 \\ 0 & 1 & 0 & 0 \\ -sin\theta_0 & 0 & cos\theta_0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0 & 0 & 0 & L_1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} cos\theta_2 & -sin\theta_2 & 0 & 0 \\ sin\theta_2 & cos\theta_2 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0 & 0 & 0 & L_2 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$T = \begin{bmatrix} c\theta_0 c\theta_1 c\theta_2 - c\theta_0 s\theta_1 s\theta_2 & -c\theta_0 c\theta_1 s\theta_2 - c\theta_0 s\theta_1 c\theta_2 & s\theta_0 & L_1 c\theta_0 c\theta_1 - L_2(s\theta_1 s\theta_2 - c\theta_0 c\theta_1 c\theta_2) \\ c\theta_1 s\theta_2 + c\theta_2 s\theta_1 & c\theta_1 c\theta_2 - s\theta_1 s\theta_2 & 0 & L_2(c\theta_1 s\theta_2 + c\theta_0 c\theta_2 s\theta_1) + L_1 c\theta_0 s\theta_1 \\ s\theta_0 s\theta_1 s\theta_2 - c\theta_1 c\theta_2 s\theta_0 & c\theta_1 s\theta_0 s\theta_2 + c\theta_0 s\theta_0 s\theta_1 & cos\theta_0 & -L_1 s\theta_0 - L_2 c\theta_2 s\theta_0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad 3.9)$$

Dimana : $c\theta_0 = cos\theta_0$; $c\theta_1 = cos\theta_1$; $c\theta_2 = cos\theta_2$

$s\theta_0 = sin\theta_0$; $s\theta_1 = sin\theta_1$; $s\theta_2 = sin\theta_2$

Kita menginginkan sebuah titik pada pusat ujung tangan (gripper), yang berarti terletak pada gripper coordinates (0,0,0). Lalu, dimana letak titik tersebut jika dilihat dari *world coordinates*? Kita dapat menghitung letak titik tersebut berdasarkan referensi *world coordinates* dengan mengalikan matrik transformasi yang telah kita dapatkan sebelumnya dengan lokasi titik pada gripper koordinat.

$$K_w = T.K_g \quad 3.10)$$

$$\begin{bmatrix} x_w \\ y_w \\ z_w \\ 1 \end{bmatrix} = T \begin{bmatrix} x_g \\ y_g \\ z_g \\ 1 \end{bmatrix} \quad 3.11)$$

$$\begin{bmatrix} x_w \\ y_w \\ z_w \\ 1 \end{bmatrix} = \begin{bmatrix} c\theta_0 c\theta_1 c\theta_2 - c\theta_0 s\theta_1 s\theta_2 & -c\theta_0 c\theta_1 s\theta_2 - c\theta_0 s\theta_1 c\theta_2 & s\theta_0 & L_1 c\theta_0 c\theta_1 - L_2(s\theta_1 s\theta_2 - c\theta_0 c\theta_1 c\theta_2) \\ c\theta_1 s\theta_2 + c\theta_2 s\theta_1 & c\theta_1 c\theta_2 - s\theta_1 s\theta_2 & 0 & L_2(c\theta_1 s\theta_2 + c\theta_0 c\theta_2 s\theta_1) + L_1 c\theta_0 s\theta_1 \\ s\theta_0 s\theta_1 s\theta_2 - c\theta_1 c\theta_2 s\theta_0 & c\theta_1 s\theta_0 s\theta_2 + c\theta_0 s\theta_0 s\theta_1 & cos\theta_0 & -L_1 s\theta_0 - L_2 c\theta_2 s\theta_0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}$$

$$\begin{bmatrix} x_w \\ y_w \\ z_w \\ 1 \end{bmatrix} = \begin{bmatrix} L_1 c\theta_0 c\theta_1 - L_2(s\theta_1 s\theta_2 - c\theta_0 c\theta_1 c\theta_2) \\ L_2(c\theta_1 s\theta_2 + c\theta_0 c\theta_2 s\theta_1) + L_1 c\theta_0 s\theta_1 \\ -L_1 s\theta_0 - L_2 c\theta_2 s\theta_0 \\ 1 \end{bmatrix} \quad 3.12)$$

Kemudian kita jabarkan dengan mengganti simbol-simbol $c\theta_0 = \cos\theta_0$,$c\theta_1 = \cos\theta_1$, $c\theta_2 = \cos\theta_2$, $s\theta_0 = \sin\theta_0$, $s\theta_1 = \sin\theta_1$, $s\theta_2 = \sin\theta_2$.

$$K_w = \begin{bmatrix} x_w \\ y_w \\ z_w \\ 1 \end{bmatrix} = \begin{bmatrix} L_1 cos\theta_0 cos\theta_1 - L_2(sin\theta_1 sin\theta_2 - cos\theta_0 cos\theta_1 cos\theta_2) \\ L_2(cos\theta_1 sin\theta_2 + cos\theta_0 cos\theta_2 sin\theta_1) + L_1 cos\theta_0 sin\theta_1 \\ -L_1 sin\theta_0 - L_2 cos\theta_2 sin\theta_0 \\ 1 \end{bmatrix} \quad 3.13)$$

Persamaan diatas merupakan persamaan forward kinematik tangan robot dengan referensi pusat koordinat bahu sebagai *world coordinates*.

3.3. Resolved Motion Rate Control Menggunakan Jacobian Inverse

Resolved Motion Rate Control (RMRC) pada compliance control yang dirancang digunakan untuk membuat trajectory awal yang diinginkan. (RMRC) menggunakan Jacobian Inverse adalah sebuah teknik untuk menyelesaikan permasalahan invers kinematik.

Invers kinematik merupakan kebalikan dari forward kinematik. Forward kinematik mentransformasikan koordinat sudut ke dalam koordinat kartesius

$$x = f(\theta) \quad 3.14)$$

Sedangkan invers kinematik mentransformasikan koordinat kartesius ke dalam koordinat sudut.

$$\theta = f^{-1}(x) \quad 3.15)$$

Resolved Motion Rate Control (RMRC) digunakan untuk mencari nilai sudut pada tiap joint pada tangan robot untuk mendapatkan posisi tertentu yang didasarkan pada posisi sebelumnya. Melalui implementasi Resolved Motion Rate Control (RMRC), maka tangan robot dapat digerakkan dengan mudah, bebas, dan fleksibel. Akan tetapi, invers kinematik tidak semudah proses forward kinematik dan banyak komputasi yang dilakukan untuk mendapatkan solusi dari permasalahan dengan akurat dan cepat.

Implementasi Resolved Motion Rate Control (RMRC) yang diterapkan pada penelitian ini didasarkan pada Jacobian Technique. Tujuan dari teknik ini adalah untuk merubah orientasi joint secara bertahap dari sebuah posisi awal yang

stabil ke arah konfigurasi state yang akan menghasilkan *end-effector* terletak pada posisi yang diharapkan didalam ruang absolute. Jumlah perubahan bertahap pada tiap iterasi di tentukan oleh hubungan parsial derivatif sudut joint (θ) dan perbedaan antara lokasi end-effector sekarang (X) dengan posisi yang diharapkan (X_d). Hubungan antara dua parameter ini akan membentuk sistem Jacobian (J). Matrik Jacobian memiliki dimensi *(m x n)* dimana *m* adalah dimensi spasial *X* dan *n* adalah ukuran dari orientasi joint yang diatur *(θ)*. Jacobian diturunkan dari persamaan 3.14 sebagai berikut.

$$dX = J(\theta)d\theta \qquad 3.16)$$

dimana

$$J_{ij} = \frac{\partial f_i}{\partial f_j} \qquad 3.17)$$

Kita tulis kembali persamaan 3.16 dalam bentuk yang mirip untuk invers kinematik (persamaan 3.15) dan hasilnya dapat dilihat pada persamaan 3.18.

$$d\theta = J^{-1}dX \qquad 3.18)$$

Bentuk diatas dapat diselesaikan menggunakan langkah iterasi. Masalahnya sekarang adalah persamaan 3.18 membutuhkan invers dari matriks Jacobian. Dimensi Matriks Jacobian kadang-kadang tidak square tergantung dari pendefinisian matrik. Oleh karena itu, biasanya digunakan psudo-inverse (persamaan 4.11) untuk menyelesaikan permasalahan matriks yang tidak square. Akan tetapi pada penelitian ini, kebetulan matrik jacobianya square 3x3 sehingga tidak perlu menggunakan pseudo invers.

$$J^{+} = J^{T}(JJ^{T})^{-1} \qquad 3.19)$$

Ketidak akurasian invers Jacobian dapat dideteksi dengan mengalikan invers Jacobian dengan original Jacobiannya kemudian kurangi matriks identitas dengan hasil tersebut. Magnitudo error dapat dihitung dengan mengalikannya dengan *dX* (persamaan 3.21). Jika besar error begitu besar, maka *dX* dapat dikurangi hingga error turun sampai batas yang dapat diterima.

Overview algoritma yang digunakan untuk mengimplementasikan solusi iterasi inverse kinematik sederhana (tanpa controller) adalah sebagai berikut:

1) Menghitung selisih antara goal position *(Xt)* dan actual position *(Xa)* dari end-effector:

$$dX = Xt - Xa$$

2) Menghitung Jacobian matrik menggunakan sudut joint sekarang. Gunakan persamaan 4.9.
3) Menghitung invers Jacobian

$$J^{-1} = \frac{adj\,(J)}{det(J)} \qquad 3.20)$$

4) Hitung error dari invers Jacobian

$$error = |\,(I - JJ^{-1})dX\,| \qquad 3.21)$$

5) Jika error > e, maka

$$dX = dX/2$$

Ulangi langkah 4

6) Hitung nilai update untuk orientasi joint dan gunakan nilai ini sebagai nilai sudut sekarang.

$$\theta = \theta + J^{I} dX \qquad 3.22)$$

7) Gunakan forward kinematik untuk menentukan apakah posisi orientasi joint end-effector yang baru sudah cukup memenuhi lokasi absolute yang diharapkan. Jika solusi belum memenuhi, maka kembali ke step 1. Jika sudah memenuhi, maka terminate program.

Komputasi invers kinematik diatas relatif membutuhkan jumlah iterasi yang cukup tinggi. Oleh karena itu, kadang perlu untuk meminimalkan jumlah degree of freedom (DOF) dari sistem untuk membuat θ sekecil mungkin.

Dari pembahasan sebelum, telah didapatkan persamaan forward kinematik tangan robot sebagai berikut (persamaan 3.13).

$$K_w = \begin{bmatrix} x_w \\ y_w \\ z_w \\ 1 \end{bmatrix} = \begin{bmatrix} L_1 cos\theta_0 cos\theta_1 - L_2(sin\theta_1 sin\theta_2 - cos\theta_0 cos\theta_1 cos\theta_2) \\ L_2(cos\theta_1 sin\theta_2 + cos\theta_0 cos\theta_2 sin\theta_1) + L_1 cos\theta_0 sin\theta_1 \\ -L_1 sin\theta_0 - L_2 cos\theta_2 sin\theta_0 \\ 1 \end{bmatrix}$$

Dengab *Kw* adalah lokasi absolute dipandang dari *world coordinates*. Setelah didapatkan forward kinematik maka dapat dicari matriks Jacobian (J) menggunakan persamaan 3.17).

$$J = \begin{bmatrix} J_{11} & J_{12} & J_{13} \\ J_{21} & J_{22} & J_{23} \\ J_{31} & J_{32} & J_{33} \end{bmatrix}$$

Dimana:

- ✓ J_{11}= - $L_1\cos(\theta_0)\sin(\theta_1)$ - $L_2(\cos(\theta_1)\sin(\theta_2) + \cos(\theta_0)\sin(\theta_1)\cos(\theta_2))$
- ✓ J_{12}= - $L_2(\sin(\theta_1)\cos(\theta_2) + \cos(\theta_0)\cos(\theta_1)\sin(\theta_2))$
- ✓ J_{13}= -$L_1\sin(\theta_0)\cos(\theta_1)$ - $L_2(\sin(\theta_0)\cos(\theta_1)\cos(\theta_2))$
- ✓ J_{21} = $L_2((-\sin(\theta_1))\sin(\theta_2) + \cos(\theta_0)\cos(\theta_2)\cos(\theta_1)) + L_1\cos(\theta_0)\cos(\theta_1)$
- ✓ J_{22}= $L_2(\cos(\theta_1)\cos(\theta_2) - \cos(\theta_0)\sin(\theta_2)\sin(\theta_1))$
- ✓ J_{23}= $L_2((-\sin(\theta_0))\cos(\theta_2)\sin(\theta_1)) - L_1\sin(\theta_0)\sin(\theta_1)$
- ✓ J_{31} = 0
- ✓ J_{32}= $L_2\sin(\theta_2)\sin(\theta_0)$
- ✓ J_{33}= $-L_1\cos(\theta_0) - L_2\cos(\theta_2)\cos(\theta_0)$

Kemudian dapat dicari invers Jacobiannya (J_{inv}) dengan menggunakan persamaan (4.12).

$$J_{inv} = \begin{bmatrix} J_{inv11} & J_{inv12} & J_{inv13} \\ J_{inv21} & J_{inv22} & J_{inv23} \\ J_{inv31} & J_{inv32} & J_{inv33} \end{bmatrix}$$

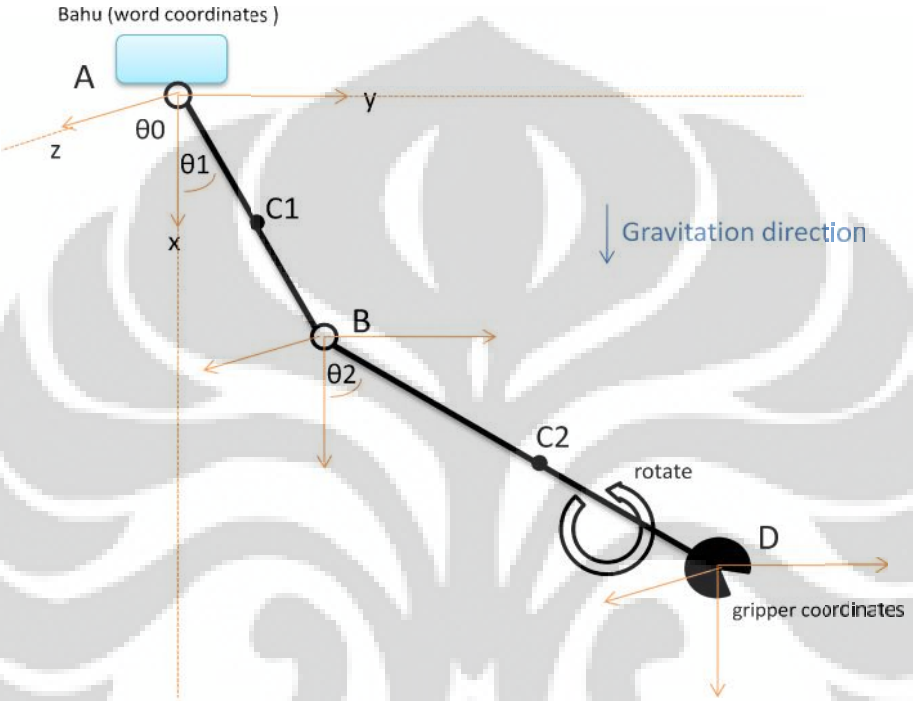
Dimana:

- ✓ det=$J_{11}(J_{22}J_{33} - J_{32}J_{23}) - J_{12}(J_{21}J_{33} - J_{31}J_{23}) + J_{13}(J_{21}J_{32} - J_{31}J_{22})$.
- ✓ $Jinv_{11}$= $(J_{22}J_{33} - J_{31}J_{23})/det$
- ✓ $Jinv_{12}$= - $(J_{12}J_{33} - J_{32}J_{13})/det$
- ✓ $Jinv_{13}$= $(J_{12}J_{23} - J_{22}J_{13})/det$
- ✓ $Jinv_{21}$= - $(J_{21}J_{33} - J_{31}J_{23})/det$
- ✓ $Jinv_{22}$= $(J_{11}J_{33} - J_{31}J_{13})/det$
- ✓ $Jinv_{23}$= - $(J_{11}J_{23} - J_{21}J_{13})/det$
- ✓ $Jinv_{31}$= $(J_{21}J_{32} - J_{31}J_{22})/det$
- ✓ $Jinv_{32}$= - $(J_{11}J_{31} - J_{31}J_{12})/det$
- ✓ $Jinv_{33}$= $(J_{11}J_{22} - J_{21}J_{12})/det$

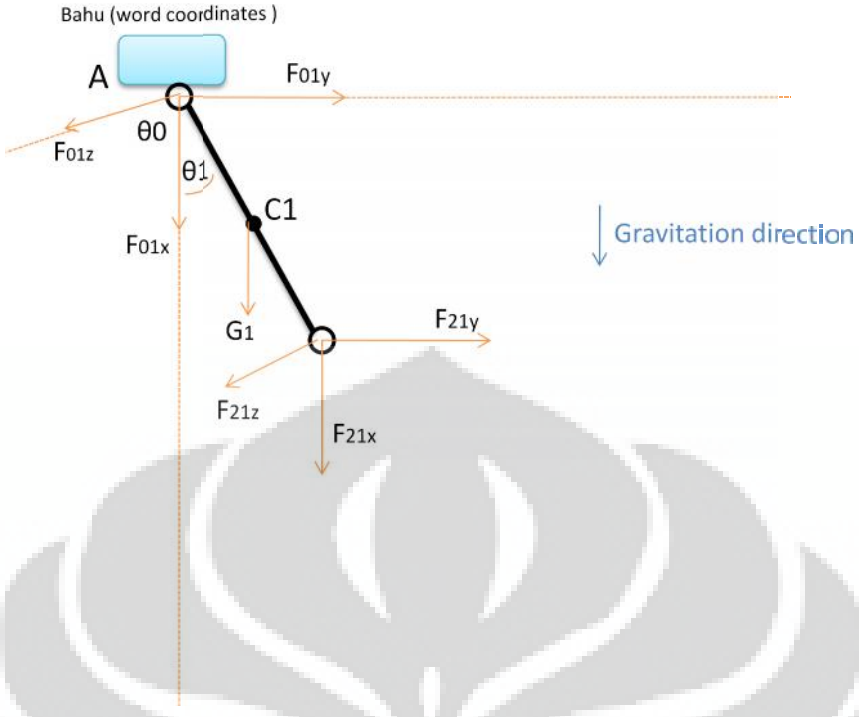
Setelah semua persamaan didapat kemudian dapat diimplementasikan compliance control dengan menambahkan controller compliance control seperti yang sudah dijelaskan pada subbab 2.1.

3.4. Newton-Euler Equation untuk Mencari Dinamik Sistem

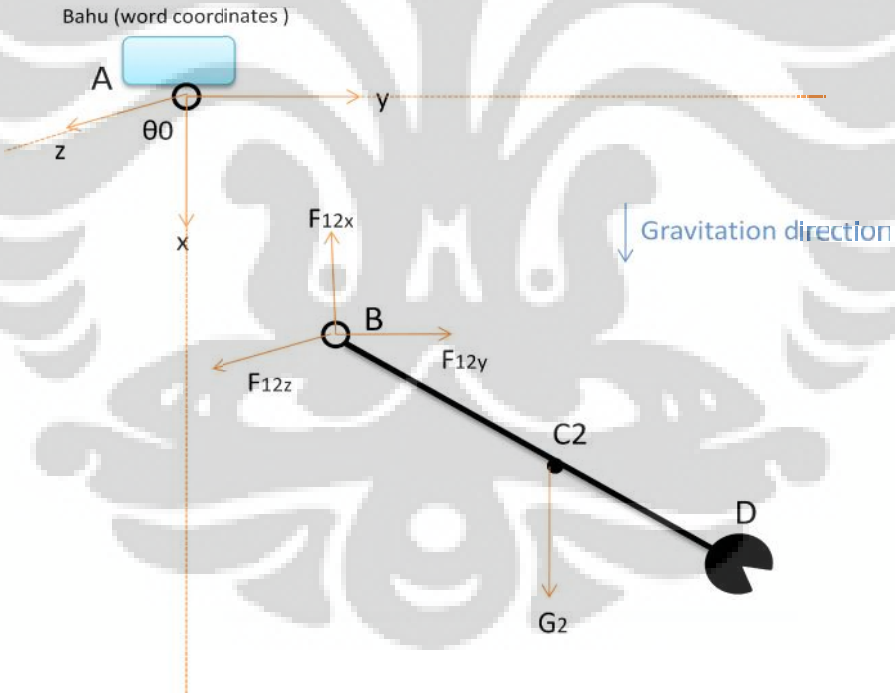
Dinamika lengan robot dapat dicari dengan menggunakan Newton/Euler Equation yang nantinya akan diiplementasikanuntuk compliance control tangan robot. Persamaan gaya/torsi yang dibebankan pada servo tiap-tiap joint dapat diturunkan dengan menerapkan metode tersebut. Kita dapat menurunkannya dengen ilustrasi gambar tangan robot berikut :



Gambar 3.7. Ilustrasi Tangan Robot



Gambar 3.8. Vektor Komponen Gaya Link 1



Gambar 3.9. Vektor Komponen Gaya Link 2

Tiap link terbuat dari bahan acylic ditambah servo dan dianggap titik pusat massanya berada ditengah (C_1 dan C_2). Jadi titik pusat massa tiap link terletak pada coordinat C_1 (x_{c1},y_{c1}, z_{c1}) dan C_2 (x_{c2},y_{c2}, z_{c2}). Sistem bergerak bebas pada

bidang vertical dan horizontal (3 dimensi). Sudut $\theta_0(t)$, $\theta_1(t)$ dan $\theta_2(t)$ dipilih dengan referensi sumbu x dan arah gravitasi searah dengan sumbu x.

Mencari persamaan Newton/Euler untuk sistem

- Link 1

Link 1 terdiri dari 2 servo, servo ke 1 rotasi terhadap sumbu y dan servo ke 2 rotasi terhadap sumbu z.

Newton/Euler untuk link 1 adalah

$m_1 a_{c1} = F_{01} + F_{21} + G_1$ 3.23)

$I_{C1}\alpha_1 = r_{C1} \times F_{01} + r_{C1B} \times F_{21}$ 3.24)

Dimana F_{01} adalah reaksi joint terhadap link 1 pada titik A, dan F_{21} adalah reaksi joint link 2 terhadap link 1 pada titik B,

$F_{01} = F_{01x}\mathbf{i} + F_{01y}\mathbf{j} + F_{01z}\mathbf{k}$, $F_{21} = F_{21x}\mathbf{i} + F_{21y}\mathbf{j}\ F_{21z}\mathbf{k}$ 3.25)

Karena link 1 memiliki titik rotasi yang tetap pada A, jumlah total momen pada titik tersebut harus sama dengan penjumlahan moment inersia dan kecepatan angularnya, maka

- Torsi dinamik Servo 1 (rotasi sumbu y)

$I_A\alpha_1 = r_{AC1} \times G_1 + r_{AB} \times F_{21}$, atau 3.26)

$$\text{Torsi} = \frac{m_1 {L_1}^2}{3}\ddot{\theta}_0 \boldsymbol{y} = \begin{vmatrix} \boldsymbol{i} & \boldsymbol{j} & \boldsymbol{k} \\ x_{c1} & y_{c1} & z_{c1} \\ m_1 g & 0 & 0 \end{vmatrix} + \begin{vmatrix} \boldsymbol{i} & \boldsymbol{j} & \boldsymbol{k} \\ x_B & y_B & z_B \\ F_{21x} & F_{21y} & F_{21z} \end{vmatrix} \text{ atau} \quad 3.27)$$

$$\frac{m_1 {L_1}^2}{3}\ddot{\theta}_0 \boldsymbol{y} = (m_1 g z_{C1} + F_{21z}\, x_B - F_{21x} z_B)\mathbf{y} \quad 3.28)$$

- Torsi dinamik Servo 2 (rotasi sumbu z)

$I_A\alpha_1 = r_{AC1} \times G_1 + r_{AB} \times F_{21}$, atau 3.29)

$$\text{Torsi} = \frac{m_1 {L_1}^2}{3}\ddot{\theta}_1 \boldsymbol{k} = \begin{vmatrix} \boldsymbol{i} & \boldsymbol{j} & \boldsymbol{k} \\ x_{c1} & y_{c1} & z_{c1} \\ m_1 g & 0 & 0 \end{vmatrix} + \begin{vmatrix} \boldsymbol{i} & \boldsymbol{j} & \boldsymbol{k} \\ x_B & y_B & z_B \\ F_{21x} & F_{21y} & F_{21z} \end{vmatrix} \text{ atau} \quad 3.30)$$

$$\frac{m_1 {L_1}^2}{3}\ddot{\theta}_1 \boldsymbol{k} = (m_1 g y_{C1} + F_{21y}\, x_B - F_{21x} y_B)\mathbf{k} \quad 3.31)$$

Dimana

x_{c1}, y_{c1}, z_{c1} = koordinat titik pusat massa link 1 terhadap world coordinate

x_B, y_B, z_B = koordinat titik ujung link 1 terhadap world coodinate

- Link 2

Link 2 hanya ada satu joint (servo 3).

Newton/Euler untuk link 2 adalah

$m_1 a_{c1} = F_{01} + F_{21} + G_1$ 3.32)

$I_{C1}\alpha_1 = r_{C1} \times F_{01} + r_{C1B} \times F_{21}$ 3.33)

Dimana $F_{12} = -F_{21}$ adalah reaksi joint link 1 terhadap link 2 pada titik B.

$m_2\ddot{x}_{C2} = F_{21x} + m_2 g$ 3.34)

$m_2\ddot{y}_{C2} = F_{21y}$

$m_2\ddot{z}_{C2} = F_{21z}$

- Torsi dinamik Servo 3 (rotasi sumbu z)

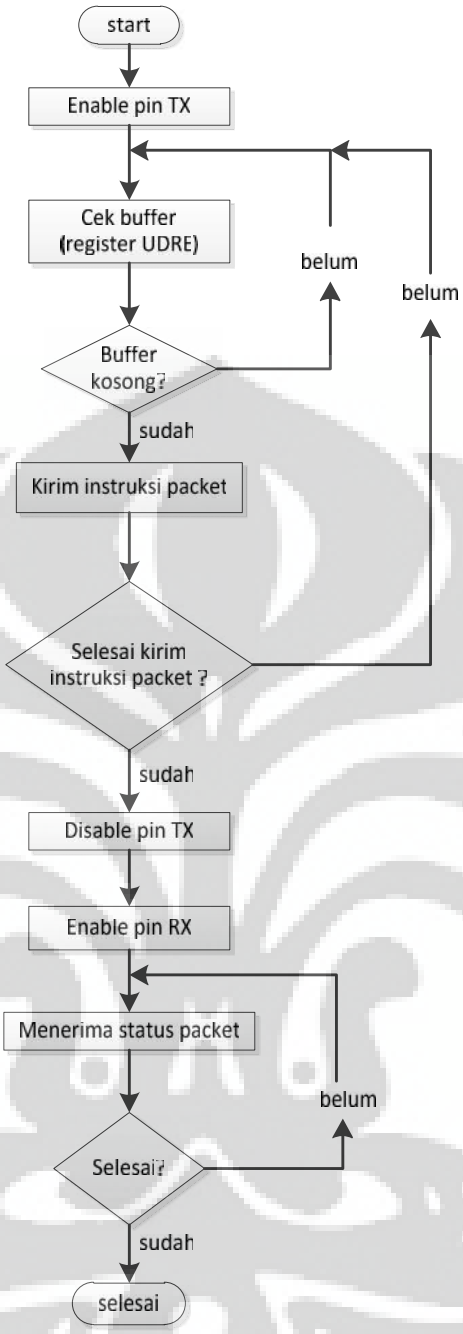
$$\frac{m_2 L_2^2}{3}\ddot{\theta}_2 \boldsymbol{k} = \begin{vmatrix} \boldsymbol{i} & \boldsymbol{j} & \boldsymbol{k} \\ x_B - x_{C2} & y_B - y_{C2} & z_B - z_{C2} \\ F_{21x} & F_{21y} & F_{21z} \end{vmatrix} \text{ atau} \quad 3.35)$$

$$\frac{m_1 L_1^2}{3}\ddot{\theta}_2 \boldsymbol{k} = (F_{21y}(x_B - x_{C2}) - F_{21x}(z_B - z_{C2}))\mathbf{k} \quad 3.36)$$

Dimana x_{c2}, y_{c2}, z_{c2} = koordinat titik pusat massa link 2 terhadap world coordinate.

3.5. ALgoritma Komunikasi antara Mikrokontroler dengan Dynamixel

Berikut algoritma protokol komunikasi komunukasi antara mikrokontroller ATmega128 dengan dynamixel RX-24F.



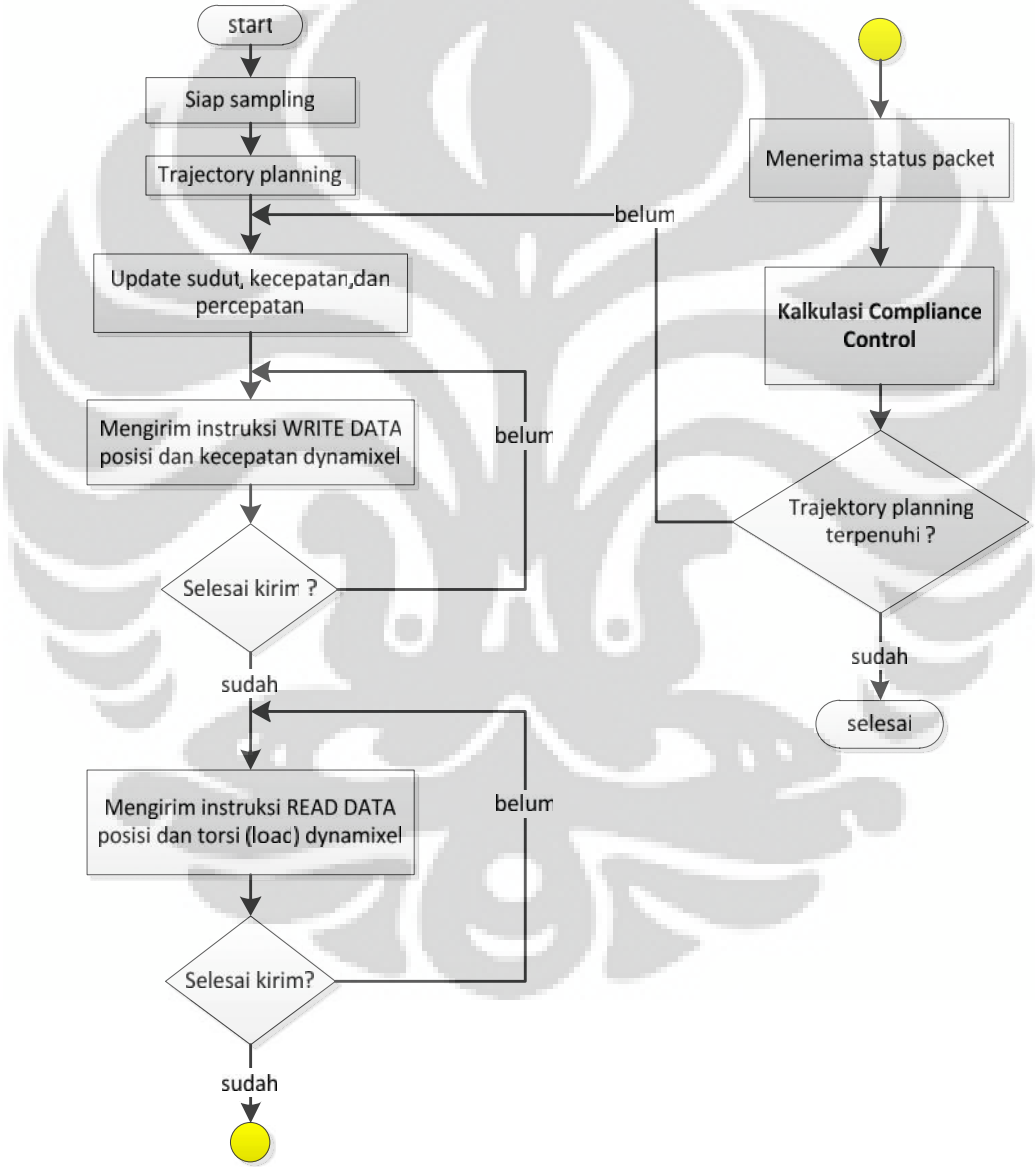
Gambar 3.10. algoritma protokol komunikasi antara mikrokontroller ATmega128 dengan dynamixel RX-24F

Kita dapat lihat dari flowchart diatas, mikrokontroller mengirim dan menerima packet data harus mengaktifkan atau menonaktifkan pin TX atau RX. Ini yang disebut komunikasi half duplek karena menerima dan mengirim data secara bergantian. Mikrokontroller mengirim data/memberi perintah ke dynamixel dengan mengaktifkan (enable) pin TX dan menonaktifkan (disable) pin RX

terlebih dahulu, kemudian mengecek buffer (register UDRE). Setelah itu mengirim instruksi yang diinginkan. Jika semua paket instruksi sudah terkirim baru disable pin TX dan enable pin RX. Kemudian menunggu status packet dari dynamixel sampai status packet diterima. Kemudian selesai.

3.6. Algoritma Compliance Control

Berikut ini merupakan algoritma program compliance control yang dirancang.



Gambar 3.11. Algoritma Compliance Control yang Dirancang

Dari gambar flowchat diatas, mikrokontroller menentukan sampling time dan trajectory planning yang diinginkan. Kemudian melalui invers kinematik,

sudut, kecepatan dan percepatan angular di update. Nilai sudut dan kecepatan ini di kirim ke dynamixel melalui packet instruksi WRITE DATA. Setelah itu mikrokontroller meminta feedback sudut dan torsi (load) dari dynamixel dengan mengirim instruksi READ DATA. Nilai feedback ini kemudian dimasukkan untuk kalkulasi compliance control yang dirancang. Compliance control ini akan mengkompensasi gaya eksternal dengan mengikuti gaya tersebut yang ditunjukkan dengan perubahan trajectory/lintasannya. Kemudian mikrokontroller mengecek apakah trajectory planning awal sudah terpenuhi atau belum. Jika sudah maka selesai dan jika belum maka sudut, kecepatan, dan percepatan akan diupdate kembali.

BAB IV
ANALISA

Setelah melakukan perancangan compliance control seperti yang telah dijelaskan pada bab III, selanjutnya adalah mencari parameter position control PD meliputi nilai Kp dan nilai Kv dengan melakukan pengujian performa dari position control yang paling optimal (nilai error minimum). Kemudian implementasi compliance control dan menguji performa dari compliance control yang dirancang. Pengujian compliance control dilakukan pada dua kondisi yaitu tanpa torsi/gaya eksternal (free motion) dan dengan torsi/gaya eksternal (pengaruh lingkungan).

4.1. Pengujian Position Control Tanpa Compliance Controller

Pada subbab ini, dilakukan pengujian position control yaitu dengan membandingkan hasil error sudut sebelum dan sesudah penambahan controller PD. Controller PD berfungsi untuk memperkecil nilai error sudut tiap joint dynamixel ketika lengan robot mengikuti trajectory planning yang sudah dibuat. Penentuan nilai Kp (gain proporsional/posisi) dan Kv (gain derivative/kecepatan) ditentukan dengan melakukan percobaan trial dan error sampai mendapatkan nilai Kp dan Kv yang optimal. Position control yang optimal perlu dilakukan terlebih dahulu sebelum penambahan compliance controller karena position control yang kurang baik akan mempengaruhi hasil dari compliance control. Jika position control ini sudah optimal, maka kompensasi torsi atau gaya ekternal hanya akan dipengaruhi oleh compliance controller yang ditambahkan. Semua torsi/gaya ekternal hanya akan dikompensasi oleh compliance controller secara penuh sehingga torsi/gaya ekternal mungkin diikuti.

4.1.1. Pengujian Tanpa Position Controller

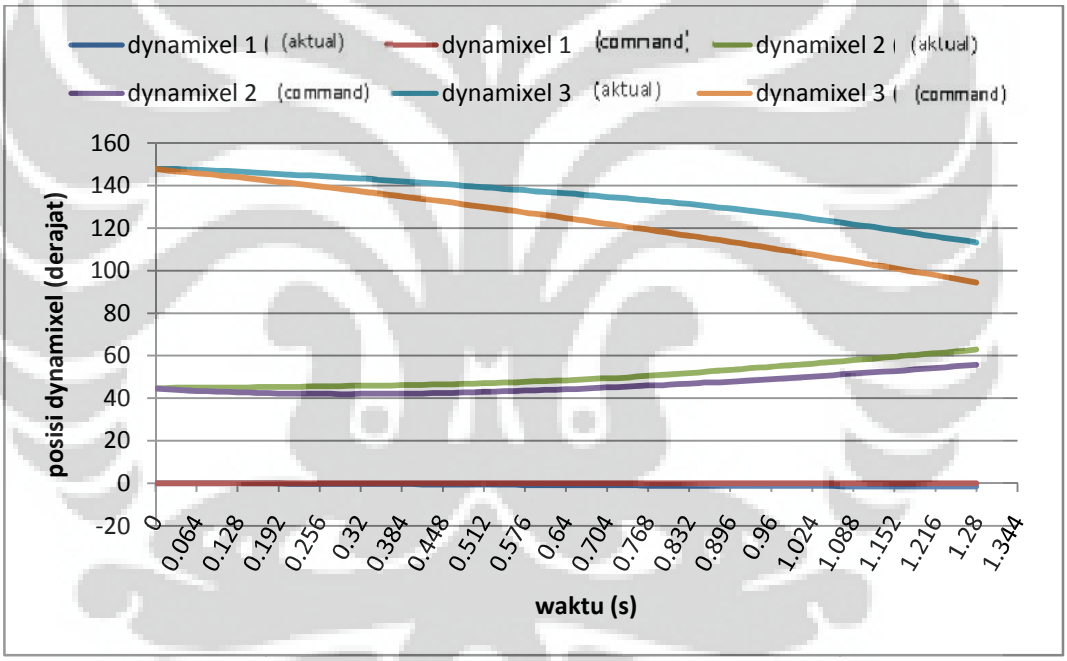
Pada percobaan ini, dilakukan pengujian tracking trajectory tanpa position controller dengan metode Inverse Jacobian biasa yang sudah dijelaskan pada BAB III subbab Resolved Motion Rate Control (RMRC). Kondisi posisi sudut update dari dynamixel dapat dirumuskan dengan persamaan

$$\theta = \theta + d\theta$$

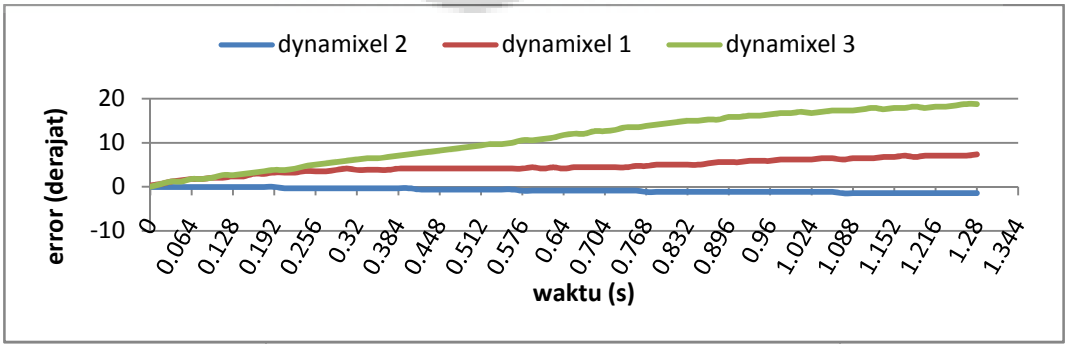
$$\theta = \theta + J^{1}dX$$

Dimana θ adalah sudut sekarang, $d\theta$ adalah perubahan sudut, J^{1} adalah Inverse Jacobian, dan dX adalah perubahan posisi trajecorty yang diset.

Percobaan ini dilakukan untuk mengetahui seberapa besar nilai error posisi sudut dynamixel saat mengikuti trajectory yang diberikan (posisi command) tanpa menggunakan position controller. Percobaan dilakukan dengan menggerakkan lengan robot sejauh 20 cm sepanjang sumbu y mulai dari koordinat (-11.3220, 7.6530, 0.0) ke koordinat (-11.3220, 27.6530, 0.0). Waktu cuplik atau sampling time yang digunakan adalah 16 ms dengan perubahan posisi tiap iterasi 2.5 mm. Hasil nilai sudut command tiap joint dan hasil nilai sudut aktual (pembacaan sensor) beserta nilai errornya adalah sebagai berikut.



Gambar 4.1. Nilai Posisi command dan Actual Dynamixel Sebelum Penambahan Position Controller



Gambar 4.2. Nilai Error Posisi Sebelum Penambahan Position Controller

Dari grafik diatas, dapa dilihat bahwa nilai error posisi dynamixel antara posisi referensi dengan pembacaan dari sensor lebih dari 6 derajat tiap iterasi trajectory hingga dapat mencapai 20 derajat bahkan lebih. Hal ini disebabkan oleh tidak adanya position control sehingga pengaruh dari luar seperti dinamika sistem dan kegagalan instruksi tidak dapat dikontrol sehingga besarnya error tinggi.

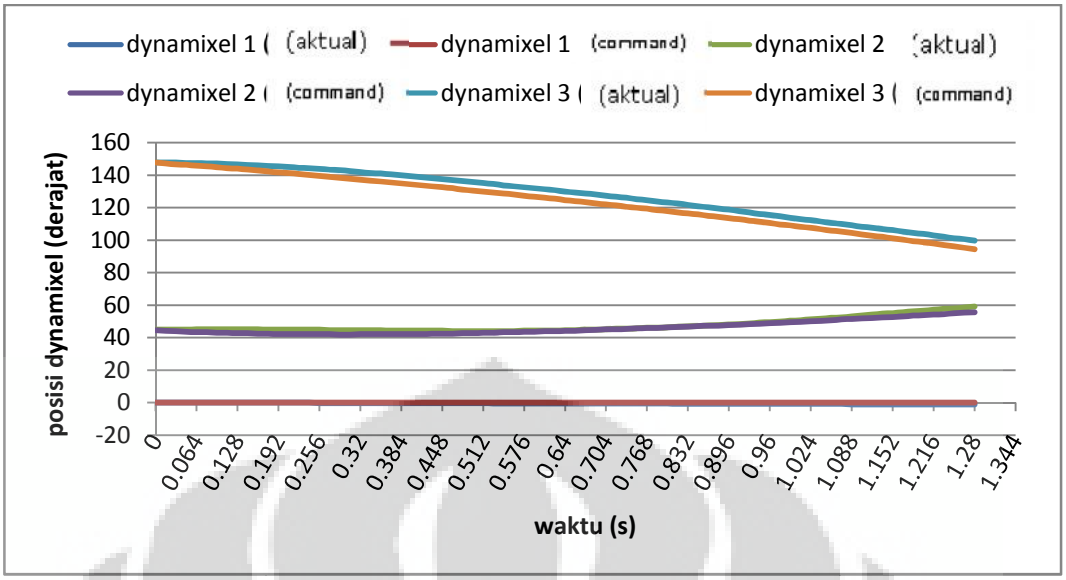
4.1.2. Pengujian Menggunakan Position Control dengan Parameter Controller yang Optimal.

Positin controller yang digunakan adalah control PD yang meliputi nilai Kp dan Kv. Position controller tersebut sudah dijelaskan pada BAB III subbab konsep dasar compliance control yang dirumuskan

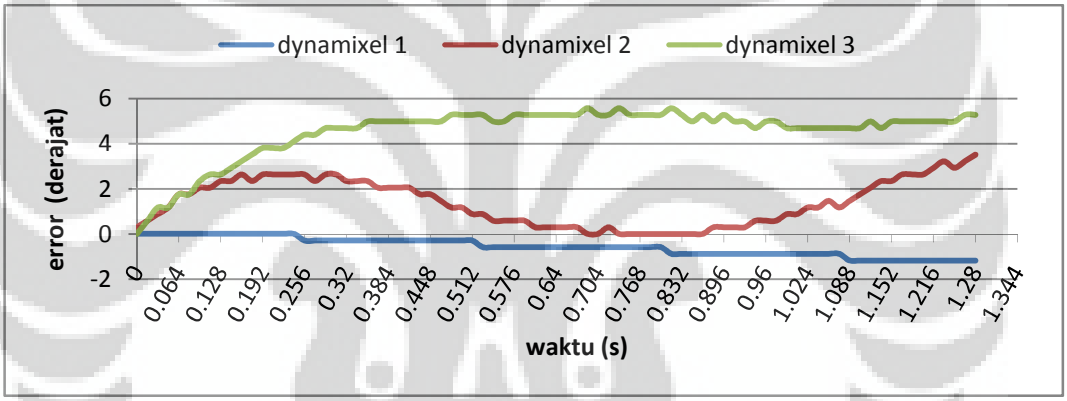
$$\ddot{\theta}^{ref} = K_p(\theta^{cmd} - \theta^{real}) + K_v\left(\dot{\theta}^{cmd} - \dot{\theta}^{real}\right)$$

Dimana $\theta^{cmd}, \dot{\theta}^{cmd}, \theta^{real}, \dot{\theta}^{real}, K_p, K_v$ adalah posisi sudut command, kecepatan sudut command, posisi sudut real (dari sensor), kecepatan sudut real (turunan posisi real), kontanta proporsional, dan konstanta derivatif/kecepatan. Hasil position controller direpresentasikan sebagai percepatan angular $\ddot{\theta}^{ref}$ yang kemudian di integralkan menjadi kecepatan angular $\dot{\theta}^{ref}$ dan dan diintegralkan lagi menjadi posisi angular θ^{ref}.

Dengan melakukan berkali-kali percobaan pada kondisi yang sama seperti percobaan tanpa kontroller, didapat nilai Kp dan Kv yang cukup optimal untuk position controller lengan robot yaitu dengan nilai Kp = 3.8 dan Kv 0.2. Hasilnya dapat dilihat pada grafik dibawah ini.



Gambar 4.3. Nilai Posisi command dan Aktual Dynamixel Setelah Penambahan Position Controller



Gambar 4.4. Nilai error sudut setelah penambahan Position Controller

Dari grafik diatas, dapat dilihat bahwa setelah penambahan position control, nilai error sudut yang didapat kurang dari 6 derajat tiap iterasi trajectory. Hal ini menunjukan bahwa dengan menambahkan position control, maka lengan robot akan berusaha mengikuti trajectory yang diberikan dengan lebih akurat dan stabil.

4.2. Pengujian Penambahan Compliance Controller

Setelah mendapatkan parameter position control yang optimal, kemudian dilakukan pengujian compliance control yang sudah dirancang. Konsep compliance control sudah dijelaskan pada BAB III subbab konsep dasar compliance control yang dapat dirumuskan

$$\ddot{\theta}^{ref} = K_p(\theta^{cmd} - \theta^{real} - K_{env}\theta_{env}^{cmd}) + K_v(\dot{\theta}^{cmd} - \dot{\theta}^{real} - D_{env}\dot{\theta}_{env}^{cmd}) - M_{env}\ddot{\theta}_{env}^{cmd}$$

Output dari compliance control dapat direpresentasikan sebagai percepatan angular $\ddot{\theta}^{ref}$ yang kemudian di integralkan menjadi kecepatan angular $\dot{\theta}^{ref}$ dan dintegralkan lagi menjadi posisi angular θ^{ref}.

Nilai torsi yang didapat dari sensor dynamixel yang merupakan torsi disturbance dilewatkan terlebih dahulu ke Low Pass Filter (LPF) untuk mendapatkan nilai torsi yang lebih smooth atau noise yang tidak terlalu besar.

Pengujian compliance control meliputi pengujian tanpa kontak luar (free motion) dan pengujian dengan kontak luar (pengaruh torsi eksternal).

4.2.1. Pengujian Tanpa Kontak Luar (Free Motion)

Pengujian tanpa kontak tidak melibatkan adanya torsi/gaya eksternal yang bekerja pada lengan robot. Oleh karena itu, torsi/gaya ekternal atau gangguan dari luar (τ_{ext}) sama dengan nol . Maka persamaan 3.3 menjadi

$$\tau_{disturbance} = \tau_{dynamic} + \tau_{ext}$$

Karena $\tau_{ext} = 0$, maka

$$\tau_{disturbance} = \tau_{dynamic}$$

Pengujian tanpa kontak (free motion) bertujuan untuk mengetahui seberapa besar tingkat error dari torsi hasil pembacaan sensor yang sudah dilewatkan ke dalam Low Pass Filter. Low Pass Filter sinyal digital dapat nyatakan dengan algoritma

α = dt / (RC + dt)

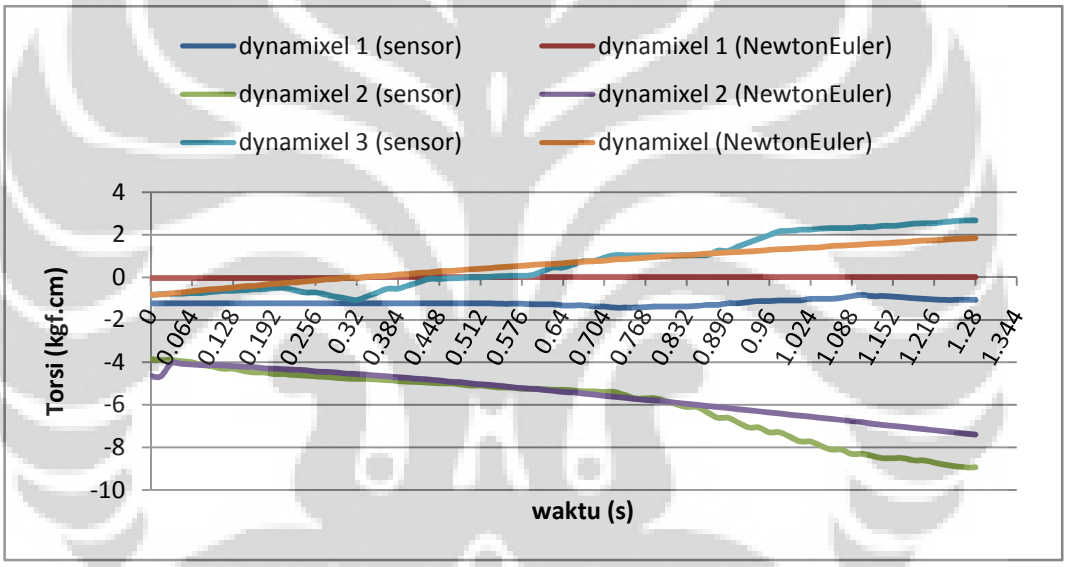
for i from 1 to n

*y[i] = y[i-1] + α * (x[i] - y[i-1])*

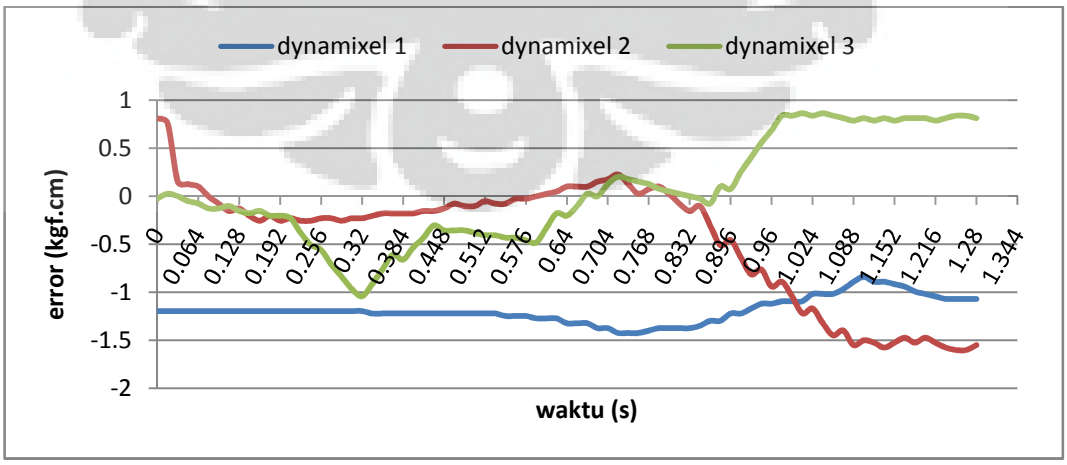
Dimana *α* adalah sampling time, *RC* adalah time konstan,(y_1,y_2,…y_n) adalah output, dan (x_1,x_2,…x_n) adalah input.

Jika nilai α kecil, maka akan membuat nilai torsi yang didapat menjadi lebih smooth tetapi membuat nilai torsi yang kurang tepat dari yang seharusnya.

Jika nilai α besar maka nilai torsi seharusnya mendekati nilai yang seharusnya, tetapi yang didapat noise besar sekali. Oleh karena itu, pada implementasi compliance control yang dirancang digunakan nilai *α* kecil dengan nilai 0.05. Pada pengukuran free motion ini, akan ditampilkan nilai error torsi yang didapat dari sensor dynamixel dengan melalui perhitungan dinamik Newton-Euler yang diakibatkan oleh berbagai factor antara lain, akibat LPF, noise, dan pengukuran yang kurang akurat yang berkaitan dengan mekanik robot meliputi massa, panjang, pusat massa, dan lain-lain. Berikut grafik nilai torsi dan error torsi pada tiap dynamixel dengan kondisi trajectory yang sama dengan pengujian position control.



Gambar 4.5. Nilai Torsi dari Sensor dan Dari Persamaan NewtonEuler



Gambar 4.6. Nilai Error Torsi Dari Sensor dan Persamaan Newton-Euler

Dari grafik diatas, dapat dilihat bahwa nilai torsi disturbance hasil pembacaan sensor dan hasil persamaan Newton-Euler pada keadaan tanpa torsi/gaya eksternal pun terdapat nilai error. nilai error yang didapatpun cukup besar. Kira-Kira bisa mencapai lebih dari 1.5 kgf.cm. Nilai ini cukup berpengaruh terhadap kompensasi gaya ekternal yang diberikan pada robot. Nilai error ini akan menambah besar error kompensasi gaya yang mungkin terjadi.

4.2.2. Pengujian Dengan kontak luar (ada gaya ekternal)

Pengujian dengan kontak luar (ada torsi/gaya eksternal) bertujuan untuk menguji performa compliance controller yang sudah dirancang dengan berbagai parameter Kf, Kenv, Denv, dan Menv untuk mendapatkan karakteristik compliance control yang dibuat dan seberapa besar kompensasi gaya yang mungkin diikuti dengan merubah trajectory mengikuti torsi/gaya tersebut. Nilai Kf, Kenv, Denv, dan Menv ditentukan dari percobaan dengan melihat performa compliance control yang cukup dapat mengkompensasi gaya ekternal. Didapat nilai parameter tersebut sebagai berikut.

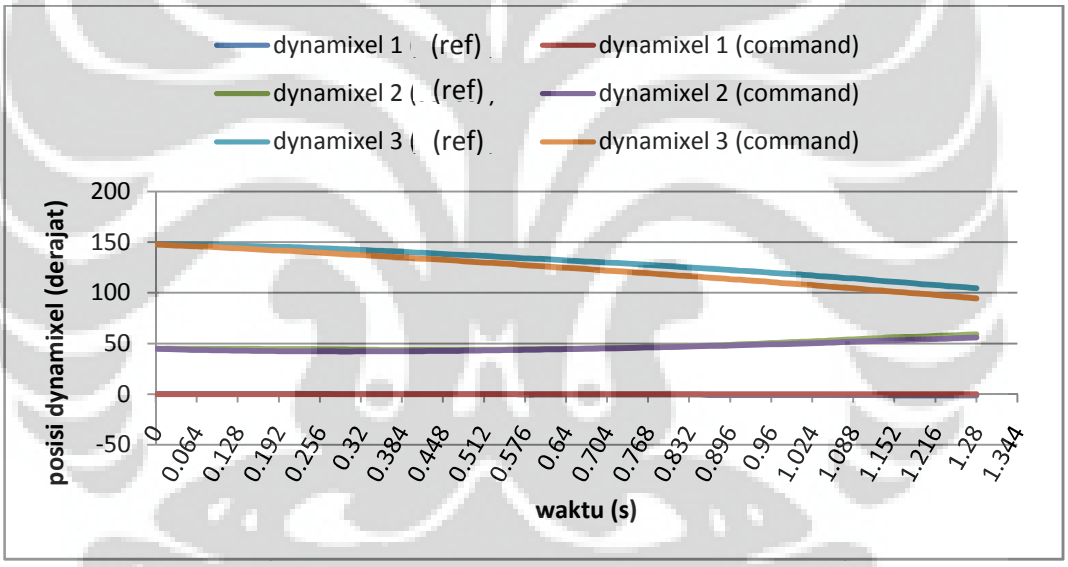
Tabel.4.1. Parameter Compliance Control

Parameter	Nilai
K_f	0.3
K_{env}	12
D_{env}	5
M_{env}	1

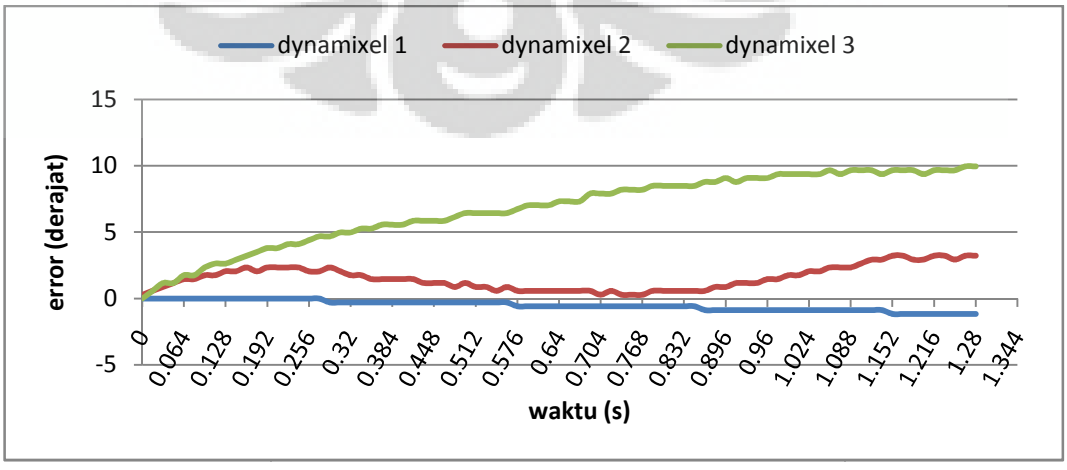
a. Tanpa Compiance Controller

Pengujian ini dilakukan untuk membandingkan antara respon sistem yang tidak ditambahkan compliance controller dengan yang ditambahkan compliance controller terhadap adannya gaya eksternal yang mungkin mungkin gaya tersebut diikuting dengan cara merubah trajectory sistem. Pengujian dilakukan dengan kondisi yang sama dengan percobaan sebelumnya, hanya saja ditambahkan sedikit tekanan ketika lengan robot berubah koordinatnya dengan arah yang berlawanan. Berikut gambar pecobaan dan hasil kompensasi trajectory yang diperoleh.

Gambar 4.7. Percobaan Compliance controller



Gambar 4.8. Nilai Posisi dari referensi dan Command tanpa compliance control

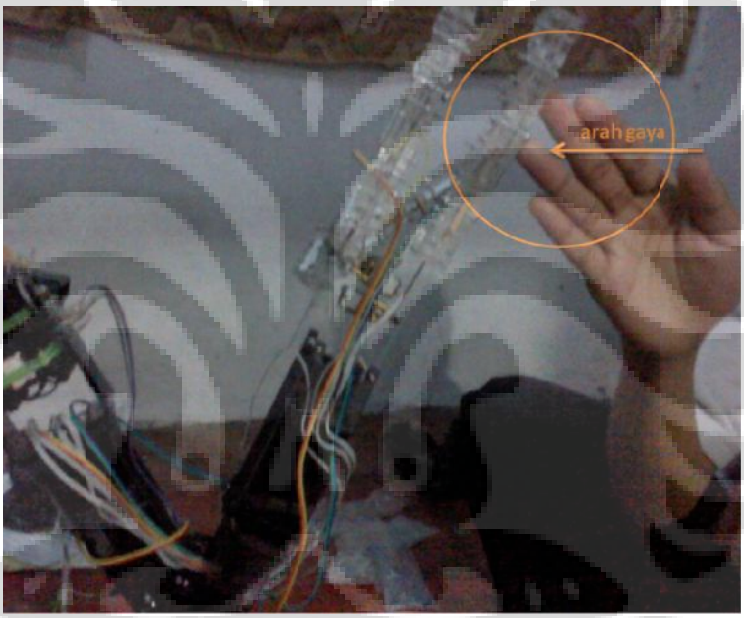


Gambar 4.9. Error Posisi Dynamixel

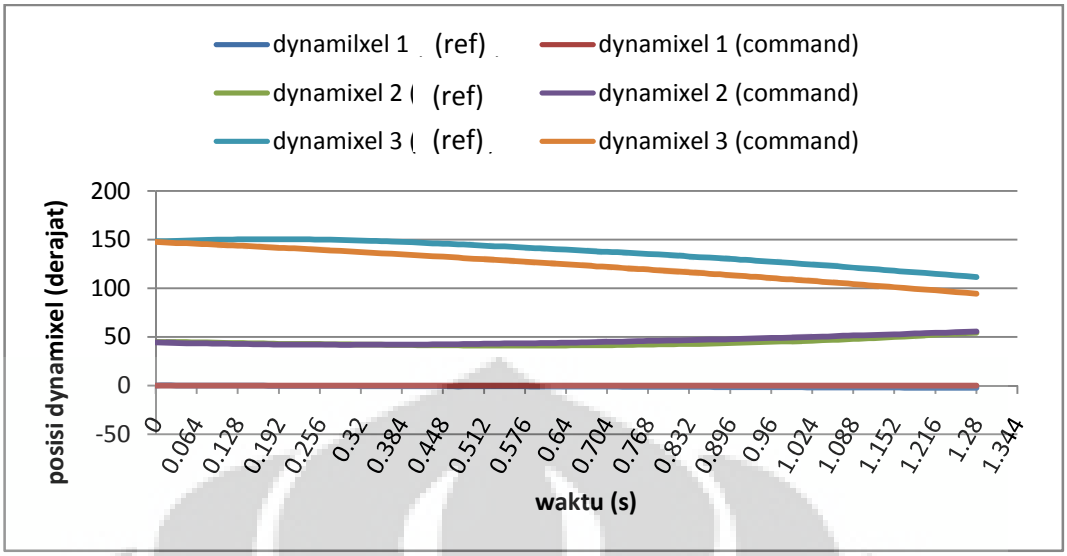
Dari grafik diatas, dapat kita lihat bahwa tanpa adanya compliance control, maka position control lah yang berperan dalam hal ini sehingga robot tidak akan mengkompensasi gaya yang mendorongnya malah akan memberikan reaksi yang kuat yang dapat menyebabkan servo rusak karena torsi yang besar untuk melawan gaya eksternalnya.

b. Dengan Compliance control

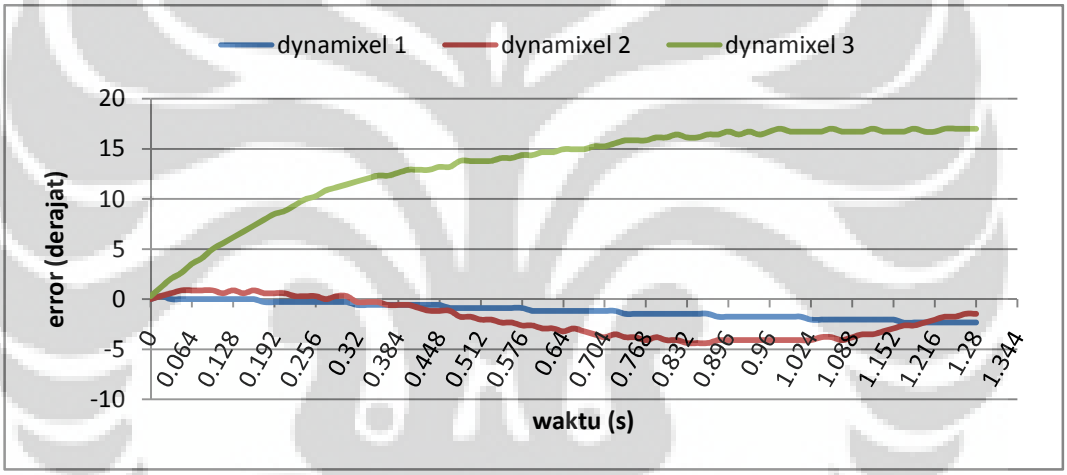
Dengan melakukan pecobaan yang sama, stelah ditambahkan compliance controller pada sistem, didapat hasil sebagai berikut.



Gambar 4.10. Percobaan Compliance controller



Gambar 4.11. Nilai Posisi Referensi dan Command dengan Compliance Control



Gambar 4.12. Error Posisi Dynamixel

Dari grafik diatas, dapat kita lihat bahwa perbedaan posisi command dengan aktualnya jauh berbeda karena ini efek dari compliance control yang mengkompensasi torsi/gaya eksternal menjadi trejektory yang mungkin torsi atau gaya tersebut diikuti. Hal ini diindikasikan dengan nilai selisih (error) yang membesar (mencapai lebih dari 15 derajat untuk dynamixel 3) antara posisi sudut command dengan sudut actual (sekarang). Walaupun pengaruhnya masih kecil belum ada kompensasi yang signifikan, paling tidak hal ini sudah menunjukkan bahwa compliance control yang di rancang sudah berhasil dan masih perlu perbaikan lagi agar hasilnya lebih baik.

BAB 5
KESIMPULAN

Setelah melakukan berbagai percobaan dan pengujian dari sistem compliance control yang dirancang, maka didapat kesimpulan sebagai berikut.

1. Compliance control yang dirancang sudah menunjukkan performa yang cukup berhasil walaupun masih kurang smooth dan terjadi error.
2. Performa compliance control juga ditandai atau diindikasikan dengan peningkatan selisih (error) antara posisi sudut command dengan posisi sudut actual hasil compliance control karena robot akan mengikuti torsi/gaya yang bekerja padanya. Hal ini menyatakan bahwa posisi control berlawanan berlawanan dengan compliance control.
3. Pengambilan data sensor yang akurat, baik untuk posisi maupun torsi sangat berpengaruh terhadap keberhasilan dari compliance controller yang dirancang dan memperkecil tingkat error sistem.
4. Position control yang baik juga akan berpengaruh pada hasil compliance controller yang ditambahkan.
5. Didapat parameter position control yang optimal yaitu nilai Kp = 3.8, Kv = 0.02 dan parameter compliance controller yang sudah cukup menunjukkan performanya, yaitu nilai Kf = 0.2, Kenv = 9, Denv = 5, dan Menv = 9. Jika nilai semua parameternya terlalu besar maka sistem menjadi tidak stabil ditunjukkan dengan adanya gerakan yang tidak menentu.

DAFTAR PUSTAKA

[1] Abdul Muis, Thesis, " Friendly Motion Control For Robot as Human Partner Over Distance". 2006. Keio University.

[2] Yudo Dewanto, Skripsi,"Rancang Bangun Bilateral Teleoperation Dengan Teknik Scaling Menggunakan PI Controller.2011. Universitas Indonesia.

[3] Prada Prempraneerach, Pasan Kulvanit. "Implementation of Resolved Motion Rate Controller with 5-Axis Robot Manipulator Arm". The First TSME International Conference on Mechanical Engineering. Oktober 2010. Thailand.

[4] Jenifer Kay. "Introduction to Homogeneous Transformation & Robot Kinematics". Rowan University Computer Science Departement. Januari 2005.

[5] Samuel R. Buss. "Introduction to Inverse Kinematics with Jacobian Transpose, Pseudoinverse and Damped Least Squares Methods". Department of Mathematics University of California, Oktober 2009. San Diego.

[6] Michael Meredith & Steve Maddock."Real-Time Inverse Kinematics: The Return oh The Jacobian". Department of Computer Science University of Sheffield. United Kingdom.

[7] Richard M. Murray, Zexiang Li, S. Shankar Sastry."A Mathematical Introduction to Robotic manipulation".

[8] Danang Junaedi."Array Multi Dimensi".

[9] Kevin M. Lynch, dkk."Collision-Free Trajectory Planning for a 3-DOF robot a Passive Joint". International Journal of Robotics Research. Agustus 2000

[10] Rederic Grupen. "Newton-Euler Equation".

[11] "Direct Dynamics Newton-Euler Equation of Motion".

[12] Raharjo, Budi. 2010. "Pemrograman C++". Bandung: Informatika Bandung.

[13] Wardhana, Lingga.2006. "Belajar Sendiri Mikrokontroler AVR Seri ATmega8535, Simulasi, Hardware, dan Aplikasi" . Yogyakarta: C.V ANDI OFFSET.

[14] Robotis. RX-24F datasheet. Juni, 2012.

[15] Atmel. Atmega128 datasheet. Juni, 2012.

LAMPIRAN

- Rancangan Program Compliance Control

a. Menyederhanakan komputasi

Menyederhanakan komputasi dengan mengumpulkan semua persamaan yang sama kemudian dikomputasi di awal dimasukkan ke dalam variable global yang nantinya dalam komputasi berikutnya tinggal dipanggil sehingga proses komputasi looping lebih cepat. Misalkan masukkan semua persamaan dalam sebuah fungsi kemudian dikomputasi di awal.

Contoh:

```
void compute(float t1,float t2,float t3,float L1,float L2)
{
s1 = sin(t1);s2 = sin(t2);s3 = sin(t3);
c1=cos(t1);c2=cos(t2);c3=cos(t3);tan1=tan(t1);tan3=tan(t3);
L1L2C = L1 + L2*c3;C1L1L2C3 = c1*(342.25 + 462.5*c3);
L1L2S3=462.5*s3;s1s2= s1*s2;c2s1=c2*s1;
}
```

b. Komputasi Newton Euler dengan menurunkan persamaan forward kinematik agar komputasi bisa lebih sederhana

Pada persamaan Neuto Euler yang dilakukan dengan penurunkan persamaan matematis didapat persamaan yang cukup panjang yang akan memperbanyak komputasi sehingga kurang efisien.

Contoh :

```
xc2_dot1 = -L2*(w1*cos(t*w1)*sin(t*w2) + w2*cos(t*w2)*sin(t*w1) +
w0*cos(t*w1)*cos(t*w2)*sin(t*w0) + w1*cos(t*w0)*cos(t*w2)*sin(t*w1) +
w2*cos(t*w0)*cos(t*w1)*sin(t*w2)) - L1*w0*cos(t*w1)*sin(t*w0) -
L1*w1*cos(t*w0)*sin(t*w1)

xc2_dot2 =w0*(L2*(a1*t*cos(t*w2)*sin(t*w0)*sin(t*w1) -
a0*t*cos(t*w0)*cos(t*w1)*cos(t*w2) + a2*t*cos(t*w1)*sin(t*w0)*sin(t*w2)) -
L1*a0*t*cos(t*w0)*cos(t*w1) + L1*a1*t*sin(t*w0)*sin(t*w1)) -
L2*(a1*cos(t*w1)*sin(t*w2) + a2*cos(t*w2)*sin(t*w1) +
a0*cos(t*w1)*cos(t*w2)*sin(t*w0) + a1*cos(t*w0)*cos(t*w2)*sin(t*w1) +
a2*cos(t*w0)*cos(t*w1)*sin(t*w2) + a1*t*w2*cos(t*w1)*cos(t*w2) +
a2*t*w1*cos(t*w1)*cos(t*w2) - a1*t*w1*sin(t*w1)*sin(t*w2) -
```

a2*t*w2*sin(t*w1)*sin(t*w2) + a1*t*w1*cos(t*w0)*cos(t*w1)*cos(t*w2) + a2*t*w2*cos(t*w0)*cos(t*w1)*cos(t*w2) - a0*t*w1*cos(t*w2)*sin(t*w0)*sin(t*w1) - a0*t*w2*cos(t*w1)*sin(t*w0)*sin(t*w2) - a1*t*w2*cos(t*w0)*sin(t*w1)*sin(t*w2) - a2*t*w1*cos(t*w0)*sin(t*w1)*sin(t*w2)) - L1*a0*cos(t*w1)*sin(t*w0) - L1*a1*cos(t*w0)*sin(t*w1) - L1*a1*t*w1*cos(t*w0)*cos(t*w1) + L1*a0*t*w1*sin(t*w0)*sin(t*w1)

Dimana : **xc2_dot1 = kecepatan linear pusat massa link 2**

xc2_dot2= percepatan linear pussat massa link 2

w1 = kepeatan angular sudut 1

w2 = kepeatan angular sudut 2

w3 = kepeatan angular sudut 3

a1 = kepeatan angular sudut 1

a2 = kepeatan angular sudut 2

a3 = kepeatan angular sudut 3

t = waktu

Jika perhitungan tersebut dilakukan didalam mikrokontroller maka akan menjadi lama. Oleh karena itu perlu melakukan sebuah cara agar komputasi tidah menjadi begitu lama.

Cara yang dilakukan adalah dengan menghitung nilai update posisi melalui forward kinematik pussat massa link 2 kemudian diturunkan menjadi kecepatan dan percepatan.

Contoh:

//forward kinematik pusat massa link 2

xc2 = L1*c1*c2 - L22*(s2*s3 - c1*c2*c3); //get x ref value (dalam cm)

dimana : xc2 = update posisi pussat massa link 2 tiap iterasi (sampling time)

//kemudian diturunkan menjadi kecepatan linear

xc2_dot1 = (xc2-xc2prev)/dt_dxl; // dt_dxl adalah sampling time (16 ms)

//kemudian diturunkan menjadi percepatan linear

xc2_dot2 = (xc2_dot1-xc2_dot1prev)/dt_dxl;

Dengan cara tersebut maka perhitungan nilai xc2_dot1 dan xc2_dot2 menjadi jauh lebih cepat dan hasilnya nilai yang didapat pun sama.