TRAJECTORY PLANNING PADA ROBOT MANIPULATOR MENGUNAKAN *INVERSE KINEMATIC*

TESIS

Disusun Oleh :

HERU SUPRAPTO

No.Pokok : 15520013

MAGISTER TEKNIK ELEKTRO

Konsentrasi : Elektronika Telekomunikasi

Tesis ini diajukan sebagai salah satu syarat untuk memperoleh gelar
Magister Bidang Ilmu Teknik Program Studi Teknik Elektro

PROGRAM PASCASARJANA
INSTITUT SAINS DAN TEKNOLOGI NASIONAL
JAKARTA
2017

PERNYATAAN KEASLIAN ISI TESIS

Saya yang bertanda tangan dibawah ini :

Nama : HERU SUPRAPTO

Nomor Pokok : 15520013

Program Studi : MAGISTER TEKNIK ELEKTRO

Konsentrasi : ELEKTRONIKA DAN TELEKOMUNIKASI

"Saya menyatakan dengan sesungguhnya bahwa Tesis ini merupakan hasil karya saya sendiri dan bukan merupakan duplikasi, serta tidak mengutip sebagian atau seluruhnya dari karya orang lain, kecuali yang telah disebutkan sumbernya.

Sepanjang pengetahuan serta keyakinan saya tidak mencantumkan tanpa pengakuan bahan-bahan yang telah dipublikasikan sebelumnya atau ditulis oleh orang lain, atau sebagian bahan yang pernah diajukan untuk gelar atau ijasah pada program pascasarjana ISTN atau perguruan tinggi lainnya.

Jakarta, 23 September 2017

Yang membuat pernyataan

Heru Suprapto

ABSTRACT

Robot manipulator is a robot arm (robot arm) in which adapted the movement of human body. Robot manipulator has two movements board on the link movement revolute joint and prismatic joint. In this case, kinematic and inverse kinematic forward becomes the control concept of robot manipulator in the working area. The working area of the kinematic forward is a joint space, while the inverse kinematic working area is cartesian space. A Robot manipulator which moved in inverse kinematic concept will be asigned in this thesis

The Robot manipulator has 2-DOF (Degree Of Freedom) motion. 2-DOF Robot Manipulator motion in inverse kinematic problem of end-effector movement is activated by determining angle of joint arm (link 1 and Link 2)

the solution inverse kinematic problem is utilited geometry approach. Movement of joints and end-effector robot is planned using trajectory planning point to point motion, Simulation result from Robot manipulator is to determined the desired end effector position. Inverse kinematic with geometry approach will result in movement where a straight-line experiment is conducted. Experimental result showed the Comparation between the measured robot movement and calculation gave a relative small error.

Keywords: robots, manipulators, *Inverse kinematic*, *trajectory planning*, *End effector*,

ABSTRAKSI

Robot manipulator merupakan lengan robot *(robot arm)* yang prinsip kerjanya diadaptasi dari cara kerja lengam manusia. *Robot manipulator* mempunyai dua gerakan pada pergerakan *link*, yaitu *revolute joint* dan *prismatic joint*. Dalam hal ini, *forward kinematic* dan *inverse kinematic* menjadi konsep kontrol daerah kerja *robot manipulator*. Daerah kerja dari *forward kinematic* berupa ruang joint, sedangkan daerah kerja *inverse kinematic* berupa ruang *cartesian*. Dalam thesis ini akan di rancang robot manipulator yang melakukan gerakan.

Robot manipulator ini memiliki pergerakan 2-DOF*(Degree Of Freedom)*, trajectory 2-DOF *Robot manipulator* dalam permasalahan *inverse kinematic* memanfaatkan pergerakan *end-effector* dengan menentukan sudut masing- masing lengan (*link 1* dan *Link 2*)

Dalam permasalahan *inverse kinematic* solusi yang digunakan adalah pendekatan *geometri*. Pergerakan sendi dan *end-effector* robot direncanakan menggunakan *trajectory planning point to point motion*, Hasil simulasi dari *Robot manipulator* adalah menentukan posisi *end effector* yang diinginkan. *Inverse kinematic* dengan pendekatan geometri akan menghasilkan pergerakan dimana dilakukan percobaan gerak garis lurus dimana hasil dari pergerakan robot akan dibandingkan hasil perhitungan.

Kata kunci: Robot, manipulator, *kinematic*, *trajectory planning, End effector*, *DOF (Degree Of Freedom),*

KATA PENGANTAR

Puji dan syukur penulis panjatkan kepada Allah SWT, karena akhirnya Tesis ini dapat diselesaikan guna memenuhi salah satu syarat dalam menyelesaikan studi pada Sekolah Pascasarjana, Program Studi Magister Teknik Elektro, Program Pascasarjana ISTN, Jakarta. Dalam proses penyelesaian Tesis ini, penulis selalu berusaha dengan sekuat kemampuan yang ada agar tidak menyimpang dari syarat-syarat yang diperlukan untuk memenuhi tuntutan ilmiahnya. Namun demikian, sebagai mahluk insani yang terbatas akan kesemestaan alam ini, penulis tidak mungkin luput dari khilaf dan kekurangan-kekurangan, maka apa yang diharapkan barangkali masih jauh dari apa yang disebut kesempurnaan penulisan.

Oleh karena itu, penulis dengan senang hati dan tangan terbuka menerima segala saran dan kritik yang bersifat konstruktif, untuk menuju ke arah perbaikan demi mencapai kesempurnaan.

Penulis menyadari bahwa tanpa bantuan dari pihak lain terutama dosen pembimbing niscaya Tesis ini tidak dapat diselesaikan dengan baik, oleh karena itu pada kesempatan ini penulis mengucapkan terimakasih kepada :

1. Dr. Ir. Abdul Multi, M.T. selaku Dekan Fakultas Teknologi Industri ISTN, Jakarta
2. Prof. Dr. Masbah R.T. Siregar, APU, selaku Ketua Program Studi Magister Teknik Elektro ISTN, Jakarta
3. Dr. Ir. Djoko Hari Nugroho, MT., selaku Dosen Pembimbing.
4. Bapak dan Ibu staf pengajar Program Studi Magister Teknik Elektro, Program Pascasarjana ISTN, Jakarta.

5. Ibu Susana selaku staf administrasi Program Studi Magister Teknik Elektro, Program Pascasarjana ISTN, Jakarta.
6. Seluruh anggota keluarga, ibu, bapak, istri, anak dan adek tercinta.
7. Keluarga besar mahasiswa Program Studi Magister Teknik Elektro, Program Pascasarjana ISTN, Jakarta, serta semua pihak yang tidak dapat disebutkan satu persatu.

Semoga dengan tersusunnya Tesis ini, dapat membawa kebaikan dan kegunaan kelak dikemudian hari.

Jakarta, September 2017

Penulis

DAFTAR ISI

BAB III PERANCANGAN ALAT DAN SIMULASI

BAB IV ANALISIS DATA DAN HASIL PENELITIAN

BAB V SIMPULAN DAN SARAN

DAFTAR GAMBAR

DAFTAR TABEL

DAFTAR PUSTAKA

1. Craig, J.J. 2005. *Introduction to Robotics Mechanics and Control*. Prentice Hall. UnitedState of America.
2. Pitowarno, Indra 2006, *robotika Desain, Control dan Kecerdasan Buatan*, Andi Offset Yogyakarta
3. *Control of Robot Manipulators,* FL Lewis, CT Abdallah, DM Dawson, 1993. This book was previously published by Prentice-Hall, Inc.
4. *Introduction to Robotics Mechanics and control Third Edition,* jhon j. Craig, 2005. Pearson prentice hail.
5. *Control of Robot Manipulators in Joint Space* - R. Kelly, V. Santibanez and A. Loria 2005
6. *Robot Motion Planning and Control* - J.P. LaumondLAAS-CNRS, Toulouse August 1997
7. *Trajectory planning and control for robot manipulations*. Ran Zhao, English2015
8. *Optimal robot trajectory planning using evolutionary algorithms*, Bhanu Kumar gouda. 2003
9. Spong, Mark W. 2007. Robot Dynamics and Control, John Wiley & Sons. New York
10. Nura, Seif Urfin. 2012. Pengendalian Lintasan End Effector Robot Lengan dengan Pendekatan Geometri Based Kinematics. Malang. Fakultas Teknik Universitas Brawijaya
11. Hamidah,Syarifah. 2008. Pengendalian Robot Lenagn dengan Menggunakan Pemrograman Visual Basic. Jurnal Ilmu Komputer Fakulatas MIPA Universitas Tanjungpura

12. Jump up^ *Paul, Richard (1981). Robot manipulators: mathematics, programming, and control : the computer control of robot manipulators. MIT Press, Cambridge, MA. ISBN 978-0-262-16082-7.*

13. J. M. McCarthy, 1990, *Introduction to Theoretical Kinematics,* MIT Press, Cambridge, MA.

14. J. M. McCarthy and G. S. Soh, 2010, *Geometric Design of Linkages,* Springer, New York.

15. Introducing visual studio 2010, david chappell. 2010

BAB I

PENDAHULUAN

1.1 LATAR BELAKANG

Robotika merupakan salah satu disiplin ilmu yang berkembang pesat seiring dengan kemajuan teknologi. Tujuan utama pembuatan robot adalah sebagai alat bantu manusia dalam menyelesaikan pekerjaan terutama pekerjaan yang memerlukan ketelitian tinggi dan berbahaya. Salah satu teknologi robotika yang banyak digunakan di industri adalah lengan robot atau yang dapat di sebut *robot manipulator*. Teknologi *robot manipulator* memiliki beberapa kelebihan untuk membantu pekerjaan manusia, yaitu mampu menyelesaikan kondisi kerja yang berulang-ulang, memiliki ketelitian tinggi dan kecepatan dalam menyelesaikan tugas serta dapat diprogram ulang sehingga mampu difungsikan untuk menyelesaikan beberapa tugas yang berbeda

Secara umum struktur *robot manipulator* dapat dibedakan menurut sumbu koordinat yang di gunakan *yaitu Cartesian, cylindrical, spherical, SCARA (Selective Compliance Assembly Robot Arm).* Robot dapat dibagi menjadi dua jenis berdasarkan struktur dan penggunaan, yaitu *mobile robot* dan *industrial robot*. *Mobile robot* memiliki kemampuan untuk bergerak tidak tetap dari suatu tempat ke tertentu. Sebaliknya, *industrial robot* biasanya terdiri dari lengan bersendi *(multi-linked manipulator)* dan *(gripper assembly)* yang melekat pada bidang yang tetap.

Teknologi robotika yang sedang berkembang ialah *Robot manipulator*. *Robot manipulator* merupakan robot yang memiliki konstruksi 2 lengan. Pada robotika, terdapat *istilah DOF (Degree of Freedom)* atau derajat kebebasan. Secara umum *DOF* pada robot adalah setiap gerakan linier atau putaran sepanjang atau sekitar pada sebuah sumbu (axis).

Salah satu dasar dari ilmu robotika adalah pemahaman mengenai kinematika dan dinamika robot. Kinematika merupakan pegetahuan atau teori tentang pergerakan objek tanpa memperhitungkan gaya-gaya yang menyebabkan benda itu bergerak.

Kinematika robot terdiri atas pergerakan rotasi dan translasi. Di dalam tinjauan kinematika gerak robot, dikenal istilah *invers kinematic* dan *forward kinematic*. Pada *forward kinematic* kita memberikan nilai pada setiap sudut motor dan mendapatkan posisi *end-effector*, sedangkan pada *invers kinematic* kita memberi masukan berupa posisi dan robot lengan akan mencari sudut pada tiap motor agar posisi *end effector* tepat sesuai dengan posisi yang diinginkan.

Sistem robot lengan pada umumnya merupakan suatu struktur mekanik yang tersusun atas beberapa batang kaku terbuat dari logam, plastik, maupun bahan lain yang sering disebut dengan *link.* Antara *link* satu dengan *link* lainnya dihubungkan oleh persendian yang disebut *joint.* Umumnya *Prismatic Joint* dan *Flat Joint* dapat menghasilkan pergeseran. Sedangkan *Shperis Joint* dan *Revolute Joint* dapat menghasilkan *Degree of Freedom (DOF)* atau derajat kebebasan . *Degree of Freedom (DOF)* atau derajat kebebasan adalah jumlah arah yang independen dimana *actuator* dari sebuah robot dapat bergerak dan menghasilkan gerakan berputar. *DOF* dapat dihitung tiap sendi dan tidak termasuk *end effector*

End effector adalah piranti yang terpasang pada lengan robot untuk melaksanakan fungsi-fungsi tertentu. *End effector* terbagi menjadi dua yaitu *gripper* dan *tool*. *End effector* dan keseluruhan bagian robot lengan bekerja pada *workspace* tertentu, tergantung kemampuan robot yang digunakan

Desain *Robot Manipulator* yang diharapkan dapat diprogram secara fleksibel oleh pengguna. Pemrograman *Robot Manipulator* membutuhkan antarmuka antara *Robot Manipulator* dengan pengguna melalui komputer**.** Pada penelitian ini, digunakan *Robot Manipulator* yang di desain dan dengan 2 derajat kebebasan dan mengunakan 2 motor stepper sebagai *actuator dan pen sebagai end effector*. Pengendalian *Robot Manipulator* dengan pendekatan *Geometry Based Kinematic.* Yaitu pendekatan kinematika yang menggunakan analisis geometri. Tujuannya penelitian adalah untuk dapat memindahkan objek pada posisi yang sesuai antara rumus perhitungan dengan gerak *Robot Manipulator*, mempermudah pengguna dalam memahami konsep kinematika gerak robot dan mempermudah dalam melakukan pemrograman menggunakan software visual studio.

1.2 RUMUSAN MASALAH

1 Permasalahan yang akan diselesaikan adalah analisis *inverse kinematics* dengan pendekatan geometris untuk mendapatkan nilai sudut pada masing-masing *joint* yang terdapat pada robot.

2 Gerak simulasi *Robot Manipulator* pada *Software visual studio* dan perhitungan *inverse kinematic* yang tepat dalam melakukan percobaan pada *Robot Manipulator*.

3 *Trajectory planning* pada simulasi *Software visual studio* dan *Robot Manipulator*

4 Gambar dua dimensi yaitu gerak garis lurus dan gerak lingkaran dengan mengunakan *Software visual studio* sebagai tampilan control robot atau *Graphical User Interface (GUI)*

1.3 BATASAN MASALAH

1. Mengunakan *software visual studio* untuk membuat *Graphical User Interface (GUI)* untuk mengerakan *Robot Manipulator DOF (Degree Of Fredom)* bergerak pada sumbu X, Y.

2. Analisis *inverse kinematics* dengan melakukan percobaan pada *robot manipulator* tangan dengan pendekatan geometris.

3. Analisis *inverse kinematics* hanya fokus pada gerak garis lurus dan gerak lingkaran dengan mengunakan *Software visual studio*

1.4 MAKSUD DAN TUJUAN PENELITIAN

tujuan utama dari studi ini:

1) Mengembangkan paket *software visual studio*, untuk menguji karakteristik dari *Robot Manipulator* dengan merancang *Graphical User Interface (GUI)* menggunakan *software Visual studio.*
2) Melakukan pengujian dengan persamaan *inverse kinematic* perbandingan antara *robot manipulator* dengan hasil perhitungan.

3) Menerapkan *Trajectory planning Point To Point Motion Robot Manipulator*.

4) Analisis *inverse kinematics* berbasis geometris diharapkan akan mendapatkan nilai sudut dari tiap joint dengan Penggunan metode kurva interpolasi polinomial diharapkan dapat menentukan trajectory sudut dari joint pada *Robot Manipulator*.

1.5 MANFAAT PENELITIAN

Mengembangkan paket perangkat lunak visual, yang mensimulasikan gerak *Robot Manipulator*. Sebuah studi yang lengkap dan analisis matematis untuk, *inverse kinematics*, *trajectory* dan masalah perencanaan lintasan (*path planning*), diimplementasikan dan diuji. Program *Graphical User Interface (GUI)* dirancang dan dikembangkan untuk mensimulasikan gerakan *Robot Manipulator* dan algoritma yang diterapkan pada lengan fisik *Robot Manipulator*. Perbandingan antara perangkat lunak yang dikembangkan dengan karakteristik gerakkan fisik *Robot Manipulator*. Penelitian ini juga membandingkan hasil dari model simulasi yang dilakukan pada *software visual studio* dengan *Robot Manipulator*

1.6 METODOLOGI PENELITIAN

Tesis ini disusun dengan cara sebagai berikut:

1. Studi Literatur

   Memberikan latar belakang teoritis, Bertujuan untuk memantapkan konsep sistem yang akan dibuat dan mempelajari lebih mendalam tentang teori *inverse kinematics* pada *robot manipulator* Mengembangkan paket *software visual studio*, untuk menguji karakteristik dari *Robot Manipulator* dengan merancang *Graphical User Interface* (GUI) menggunakan software Visual studio.

2. Perancangan *Robot Manipulator*

   Perancangan *Robot Manipulator* dengan desain menyesuaikan dengan robot gunakan di industri dengan perhitungan analisis inverse kinematics berbasis geometris untuk mendapatkan sudut pada joint. Hasil perhitungan sudut akan menjadi input pada proses gerak dari robot.

3. Perancangan Implementasi

   Melakukan implementasi dari hasil analisis inverse kinematics kedalam robot Melakukan pengujian dengan persamaan inverse kinematic perbandingan antara *robot manipulator* dengan hasil perhitungan.

4. Analisis

   inverse kinematics berbasis geometris diharapkan akan mendapatkan nilai sudut dari tiap joint dengan melakukan percobaan garis lurus diharapkan dapat menentukan trayektori sudut dari joint pada *Robot Manipulator.*

5. Hasil pengujian sistem yang dibahas. kesimpulan umum yang disediakan serta rekomendasi untuk pekerjaan di masa yang akan datang

1.7 SISTEMATIKA PENULISAN

Untuk mendapatkan suatu cara yang teratur dan sistematik maka secara garis besar penulis membagi pembahasan tesis ini menjadi 5 bab, dengan urutan sebagai berikut :

Bab I. Pendahuluan mencakup tentang latar belakang masalah, rumusan masalah, batasan masalah, maksud tujuan penelitian, manfaat penelitian, metodologi penelitian, dan sistematika penulisan.

Bab II. Tinjauan pustaka berisikan tinjauan pustaka hasil bacaan, kerangka pemikiran, dan hipotesis.

Bab III. Perancangan dan pembuatan piranti berisikan metode dan desain perancangan.

Bab IV. Hasil dan Pembahasan, berisikan data, analisa perhitungan, analisa data dan pengujian / pengukuran piranti.

Bab V. Kesimpulan

Lampiran

BAB II

TINJAUAN PUSTAKA

2.1 Pengertian Robot

Kata robot diambil dari bahasa Ceko (Chech), yang memiliki arti pekerja (*worker*). Robot merupakan suatu perangkat mekanik yang mampu menjalankan tugas-tugas fisik, baik di bawah kendali dan pengawasan manusia, ataupun yang dijalankan dengan serangkaian program yang telah didefinisikan terlebih dahulu atau kecerdasan buatan (artificial intelligence). Ada banyak defenisi yang dikemukakan oleh para ahli mengenai robot. Beberapa ahli robotika berupaya memberikan beberapa defenisi, antara lain (Gonzalez, 1987) :

1. Robot adalah sebuah manipulator yang dapat di program ulang untuk memindahkan tool, material, atau peralatan tertentu dengan berbagai program pergerakan untuk berbagai tugas dan juga mengendalikan serta mensinkronkan peralatan dengan pekerjaannya, oleh Robot Institute of America.
2. Robot adalah sebuah sistem mekanik yang mempunyai fungsi gerak analog untuk fungsi gerak organisme hidup, atau kombinasi dari banyak fungsi gerak dengan fungsi intelligent, oleh official Japanese. Industri robot dibangun dari tiga sistim dasar (Eugene, 1976):

2.1.1. Bagian Robot

1. Struktur mekanis, yaitu sambungan-sambungan mekanis (link) dan pasangan-pasangan (joint) yang memungkinkan untuk membuat berbagai variasi gerakan.
2. Sistem kendali, dapat berupa kendali tetap (fixed) ataupun servo, yang dimaksud dengan sistem kendali tetap yaitu suatu kendali robot yang pengaturan gerakannya mengikuti lintasan (path), sedangkan kendali servo yaitu suatu kendali robot yang pengaturan gerakannya dilakukan secara point to point (PTP) atau titik pertitik.

3. Unit penggerak (actuator), seperti *hydrolic, pneumatic, electric* ataupun kombinasi dari ketiganya, dengan atau tanpa sistim transmisi. Torsi (force) dan kecepatan yang tersedia pada suatu aktuator diperlukan untuk mengendalikan posisi dan kecepatan. Transmisi diperlukan untuk menggandakan torsi. Seperti diketahui menambah torsi dapat menurunkan kecepatan, dan meningkatkan inersia efektif pada sambungan. Untuk mengurangi berat suatu sistem robot maka aktuator tidak ditempatkan pada bagian yang digerakkan, tetapi pada sambungan yang sebelumnya.

Ada beberapa jenis transmisi yang banyak dipakai, antara lain *Belt, Cable, Chain* dan roda gigi. Jika sebelumnya robot hanya dioperasikan di laboratorium ataupun dimanfaatkan untuk kepentingan industri, di negara-negara maju perkembangan robot mengalami peningkatan yang tajam, saat ini robot telah digunakan sebagai alat untuk membantu pekerjaan manusia. Seiring dengan berkembangnya teknologi, khususnya teknologi elektronik, peran robot menjadi semakin penting tidak saja dibidang sains, tapi juga di berbagai bidang lainnya, seperti di bidang kedokteran, pertanian, bahkan militer. Secara sadar atau tidak, saat ini robot telah masuk dalam kehidupan manusia sehari-hari dalam berbagai bentuk dan jenis. Ada jenis robot sederhana yang dirancang untuk melakukan kegiatan yang sederhana, mudah dan berulang-ulang, ataupun robot yang diciptakan khusus untuk melakukan sesuatu yang rumit, sehingga dapat berperilaku sangat kompleks dan secara otomatis dapat mengontrol dirinya sendiri sampai batas tertentu. Robot memiliki berbagai macam konstruksi. Diantaranya adalah:

2.2 *Robot manipulator*

2.2.1. **Klasifikasi *Robot manipulator***

Secara umum struktur robot dapat dibedakan menurut lengan koordinat yang digunakan, untuk lebih jelasnya diuraikan dalam tabel 2.1

Tabel 2.1 Klasifikasi *Robot manipulator*

NO	Jenis robot	Lengan 1	Lengan 2	Lengan 3	Total Rotasi
1	Cartesian	P	P	P	0
2	Cylindrical	R	P	P	1

3	Spherical	R	R	P	2
4	SCARA	R	R	P	2
5	Articulated	R	R	R	

Catatan : P=Prismatic joint yaitu pergeseran sepanjang lengan tertentu

R= Revolute joint yaitu perputaran pada lengan tertentu.

SCARA adalah *Selective Compliant Assembly Robot Arm*. Pertama kali dibuat oleh perusahaan USA bernama Adept pada 1984. Sistem penggerak SCARA robot tersambung secara langsung pada lengan tanpa gear atau sistem belt, sehingga membuat mekanisme gerakannya bekerja cepat, simple namun tetap akurat. *SCARA ROBOT* ini banyak digunakan sebagai robot aseembly part dengan ukuran yang kecil degan kecepatan sedang. Keberhasilan robot ini dapat dibuat karena utama untuk faktor-faktor berikut:

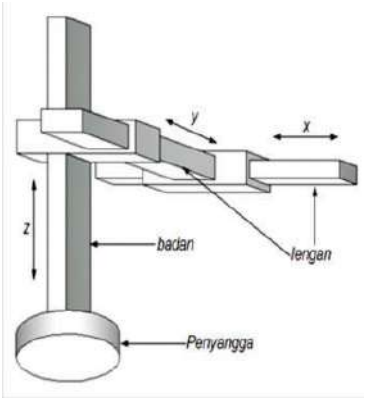
- Presisi, kecepatan tinggi, karena struktur ringan
- Dimensi kecil, gerakan halus
- Sederhana dan dapat diandalkan struktur
- Kemudahan instalasi dan penggunaan

2.2.2. Konfigurasi Robot

Konfigurasi robot adalah pola susunan *link* dan *Joint* yang menghasilkan karateristik gerakan tertentu. Secara umum konfigurasi robot dibagi menjadi beberapa jenis tergantung pada *Joint* yang digunakan, antara lain:

2.2.2.1. Konfigurasi Cartesian

Struktur Robot ini terdiri dari tiga lengan linier (*prismatic*). Masing-masing lengan dapat bergerak kearea lengan x-y-z. Keuntungan robot ini adalah pengontrolan posisi yang mudah dan mempunyai struktur yang lebih kokoh.

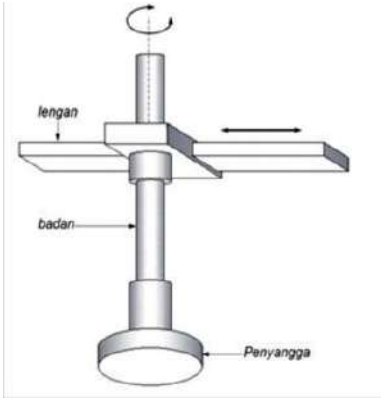


Gambar 2.1 Konfigurasi Cartesian.

Pada Gambar 2.1 memperlihatkan manipulator berkonfigurasi kartesian dimana secara relatif adalah yang paling kokoh untuk tugas mengangkat beban yang berat. Struktur ini banyak dipakai secara permanen pada instalasi pabrik baik untuk mengangkat dan memindah barang-barang produksi maupun untuk mengangkat peralatan-peralatan berat pabrik ketika melakukan kegiatan instalasi.

2.2.2.2. Konfigurasi Cylindrical

Struktur dasar dari robot Cylindrical adalah terdiri dari *horisontal arm* dan *vertical arm* yang dapat berputar pada dasar landasannya, dan dapat dilihat pada Gambar 2.2. Jika dibandingkan dengan robot kartesian, robot silindris mempunyai kecepatan gerak lebih tinggi dari *end effector*-nya, tapi kecepatan tersebut tergantung momen inersia dari beban yang dibawanya.

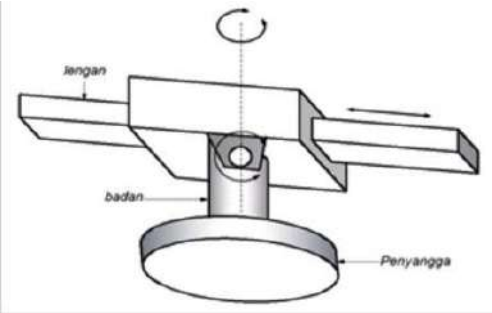


Gambar 2.2 Konfigurasi silinder.

Konfigurasi silinder mempunyai kemampuan jangkauan berbentuk ruang silinder yang lebih baik, meskipun sudut ujung lengan terhadap garis penyangga tetap. Konfigurasi ini banyak diadopsi untuk sistem gantry atau crane karena strukturnya yang kokoh untuk tugas mengangkat beban.

2.2.2.3. Konfigurasi Spherical

Konfigurasi struktur robot ini mirip dengan sebuah tank dimana terdiri atas *rotary base*, *elevated pivot*, dan *telescopic*. Keuntungan dari robot jenis ini adalah fleksibilitas mekanik yang lebih baik.

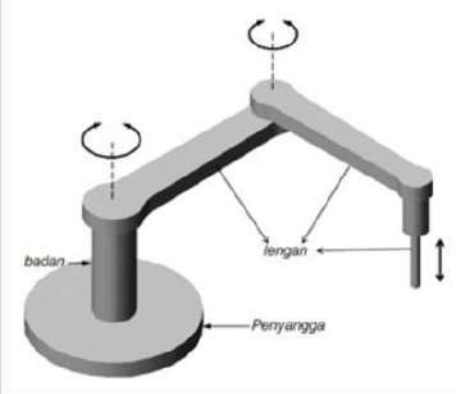


Gambar 2.3 Konfigurasi Spherical.

Pada Gambar 2.3 terlihat konfigurasi Spherical dimana badan dapat berputar ke kiri atau kanan. Sendi pada badan dapat mengangkat atau menurunkan pangkal lengan secara polar. Lengan ujung dapat digerakkan maju-mundur secara translasi

2.2.2.4. Konfigurasi SCARA (*Selective Compliance Assembly Robot Arm*)

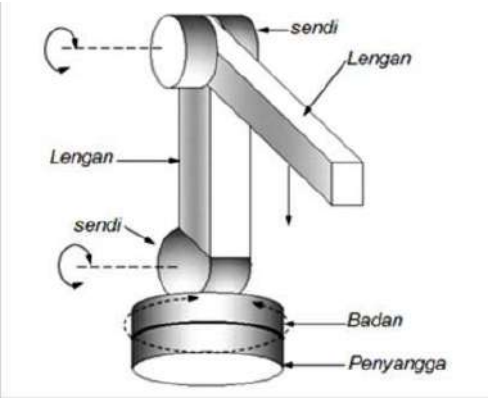
Robot *assembly* bisa didesain menurut koordinat kartesian, silindris maupun spheris. Pada beberapa aplikasi hanya membutuhkan lengan gerak vertikal, misalnya robot *assembly* yang memasang komponen pada PCB. Robot ini mempunyai lengan dengan dua artikulasi, sedangkan *wrist* mempunyai gerakan *linier* dan *rolling*. Struktur robot *assembly* dapat dilihat pada Gambar 2.4.



Gambar 2.4 Struktur robot SCARA.

2.2.2.5. Konfigurasi Articulated

Robot ini terdiri dari tiga lengan yang dihubungkan dengan dua *revolute Joint*. *elbow Joint* menghubungkan *force arm* dengan *upper arm*. *Shoulder Joint* menghubungkan *upper arm* dengan *base*. Struktur robot artikulasi ini dapat dilihat pada gambar 2.5.



Gambar 2.5 Konfigurasi Articulated.

Konfigurasi ini yang paling populer untuk melaksanakan fungsi layaknya pekerja pabrik seperti mengangkat barang, mengelas, memasang komponen mur dan baut, dan sebagainya. Struktur lengan-sendi cocok digunakan untuk menjangkau daerah kerja yang sempit dengan sudut jangkauan yang beragam.

2.2.3. End Effector

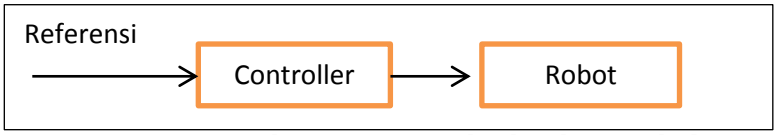
Kemampuan robot juga tergantung pada piranti yang dipasang pada lengan robot. Piranti ini biasanya dikenal dengan *end effector*. E*nd effector* ada dua jenis yaitu Pencengkram (*griper*) yang digunakan untuk memegang dan menahan obyek, peralatan (*tool*) yang digunakan untuk melakukan operasi tertentu pada suatu obyek. Contohnya: bor, proses pengecatan, gerinda, las dan sebagainya.

2.2.4 Sistem Kendali Robot

1. **Sistem kendali** dapat dikatakan sebagai hubungan antara komponen yang membentuk sebuah konfigurasi sistem, yang akan menghasilkan tanggapan sistem yang diharapkan. Jadi harus ada yang dikendalikan, yang merupakan suatu sistem fisis, yang biasa disebut dengan kendalian *(plant).*
2. **Masukan dan keluaran** merupakan variabel atau besaran. Keluaran merupakan hal yang dihasilkan oleh kendalian, artinya yang dikendalikan; sedangkan masukan adalah yang mempengaruhi kendalian, yang mengatur keluaran. Kedua dimensi masukan dan keluaran tidak harus sama.

Pada sistem kendali dikenal sistem terbuka *(open loop system)* dan sistem tertutup *(closed loop system).* Sistem kendali terbuka atau umpan maju *(feedforward control)* umumnya mempergunakan pengatur *(controller)* serta aktuator kendali *(control actuator)* yang berguna untuk memperoleh respon sistem yang baik. Sistem kendali ini keluarannya tidak diperhitungkan ulang oleh *controller*. Suatu kondisi apakah benar-benar telah mencapai target seperti yang dikehendaki masukan atau referensi, tidak dapat mempengaruhi kinerja *Controller*.

Diagram loop terbuka atau umpan maju *(feed forward control)* dapat dinyatakan dalam gambar berikut ini.

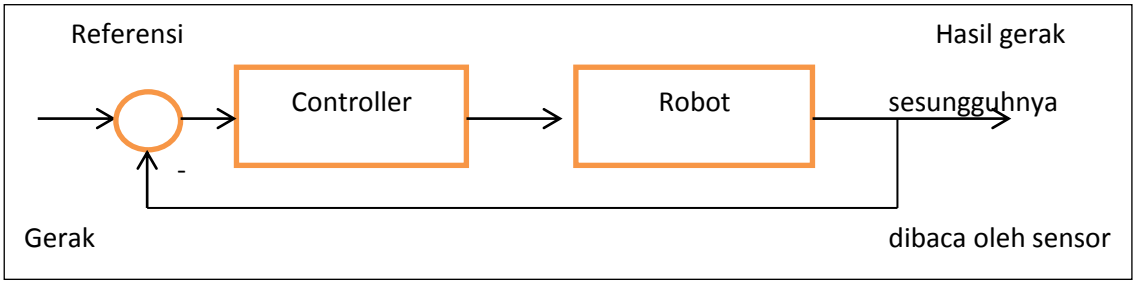


Gambar 2.6. Kontrol robot loop terbuka

Sumber : Pituwarno (2006)

Kontrol loop terbuka atau umpan maju (*feedforward control*) dapat dinyatakan sebagai sistem kontrol yang outputnya tidak diperhitungkan ulang oleh kontroller. Keadaan apakah robot telah benar-benar mencapai target seperti yang dikehendaki sesuai referensi, adalah tidak mempengaruhi kerja kontroller.

Kontrol robot loop tertutup dapat dinyatakan seperti gambar di bawah:



Gambar 2.7. Kontrol robot loop tertutup

Sumber : Pituwarno (2006)

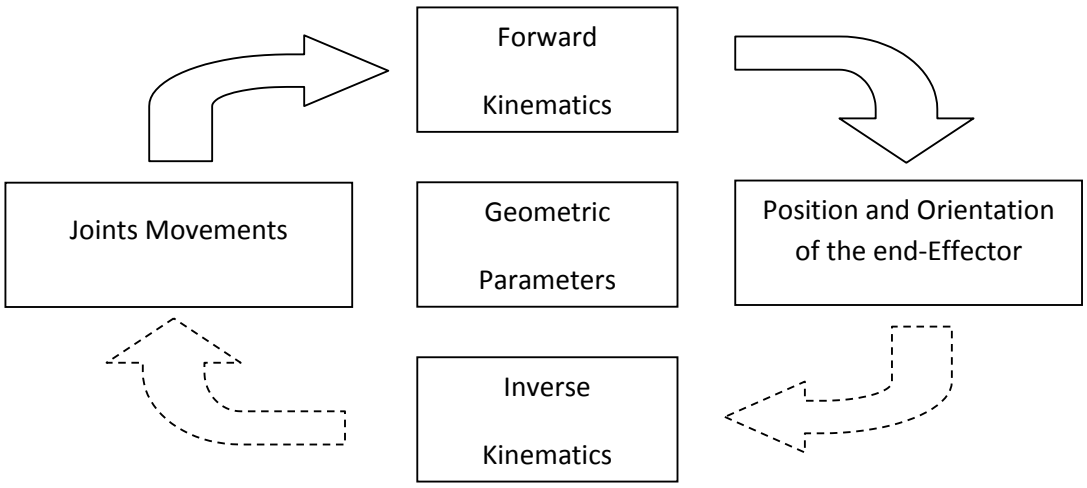
Pada gambar di atas , jika hasil gerak aktual telah sama dengan referensi maka input kontroler akan sama dengan nol. Artinya kontroler tidak lagi memberikan sinyal aktuasi kepada robot karena target akhir perintah gerak telah diperoleh. Makin kecil error terhitung maka makin kecil pula sinyal pengemudian kontroler terhadap robot, sampai pada akhirnya mencapai kondisi tenang (steady state).

2.3 **Kinematics**

Fu,K.S.,R.C. gonzales,C. S. G. Lee 1987. Robotics: Control, Sensing, vision, and intelegence, first edition bahwa kinematika adalah ilmu tentang gerak tanpa memperhatikan penyebabnya salah satunya adalah gaya yang mempengaruhinya, berhubungan dengan geometri dari gerakan. Dalam mengkaji kinematik perlu dilakukan deskripsi analisis dari penempatan posisi secara spasial dari lengan robot sebagai sebuah fungsi waktu. Secara garis besar, kinematika ini membahas tentang hubungan antara derajat kebebasan masing - masing *Joint*, posisi, serta orientasi dari

end –effector pada lengan robot''. Terdapat dua topik pembahasan mendasar pada kinematik lengan robot.

1. *Direct atau forward kinematics.*
2. *Invers kinematics atau arm solution.*



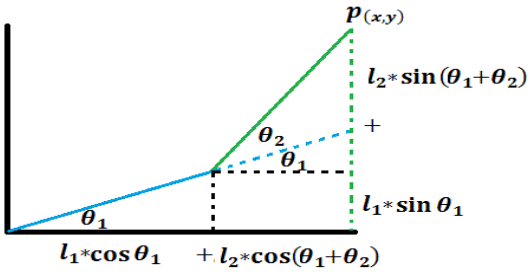
Gambar 2.8 Forward and inverse kinematics diagram

2.3.1 Forward Kinematics

Untuk mengendalikan sebuah robot, khususnya manipulator, diperlukan sebuah kontrol untuk mengendalikan tangan atau kaki. Pengendalian robot ini disebut dengan forward kinematics. Forward Kinematics adalah suatu fungsi yang memetakan sudut terhadap posisi (x,y,z). Pengendalian ini menggunakan beberapa parameter

2.3.1.1. Forward kinematic Robot 2 DOF (degre of freedom)

Semua sumbu z pada gambar tidak ditampilkan karena tegak lurus dengan arah pandang kita. Dari gambar diatas kita tetapkan frame dasar-nya (*base frame*) adalah $o_0,z_0y_0x_0$. Titik origin O_0 kita tentukan pada titik dimana sumbu z0 pada pangkalnya (bayangkan titik z_0 menembus kedalam halaman ini), dan penentuan x0 dan y0 mengikuti aturan tangan kanan seperti pada gambar.



$$x = l_1 * \cos\theta_1 + l_2 * \cos(\theta_1 + \theta_2)$$
$$y = l_1 * \sin\theta_1 + l_2 * \sin(\theta_1 + \theta_2)$$

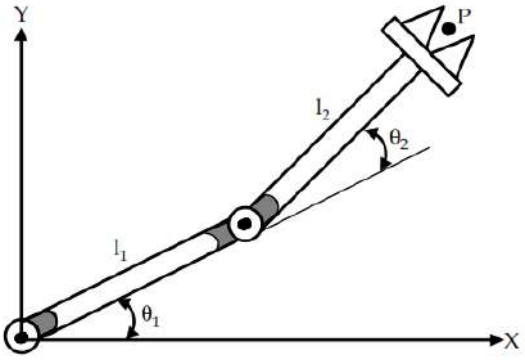
2.3.2 Iverse Kinematics

Invers kinematika (IK) analisis menentukan sudut bersama untuk posisi diinginkan dan orientasi di ruang Cartesian. Total transformasi matriks dalam persamaan. (Diatas) akan digunakan untuk menghitung invers kinematika persamaan. IK adalah masalah yang lebih sulit daripada kinematika maju.

Solusi invers kinematis lebih kompleks daripada kinematika langsung dan tidak ada metode solusi analitik global. Manipulator setiap kebutuhan metode tertentu mengingat struktur sistem dan pembatasan. Ada dua pendekatan solusi yaitu geometris dan aljabar digunakan untuk berasal solusi kinematika invers. Mari kita mulai dengan pendekatan geometris

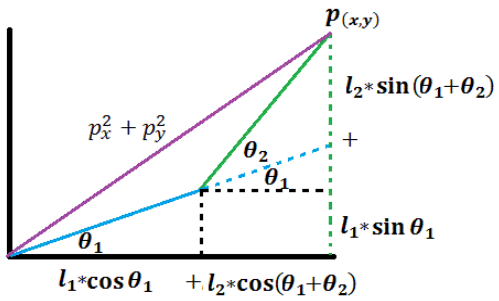
2.3.2.1. *Metode Geometric*

Pendekatan yang digunakan Endra pitowarno dalam mengananlisis *permasalahan inverse kinematics* pada persamaan *kinematic* lengan robot juga mengunakan metode geometri yang di tulis berdasarkan hukum trignometri. Penyelesaian *masalah inverse kinematic* berdasarakan pendekatan trigonometri dapat digunakan untuk analisis Kinematik robot tangan sendi



Gambar 2.9 *kinematic Robot manipulator*

Inverse kinematic 2 DOF (*degre of freedom*)



Inverse kinematic robot 2 DOF dapat di jabarkan mengunakan hukum identitas trigonometri

1. berdasarkan rumusTrigonometri

$$p_x = l_1 c\theta_1 + l_2 c\theta_{12}$$
$$p_y = l_1 s\theta_1 + l_2 s\theta_{12}$$

Dimana $c\theta_{12} = c\theta_1 c\theta_2 - s\theta_1 s\theta_2$ and $s\theta_{12} = s\theta_1 c\theta_2 + c\theta_1 s\theta_2$

2. *theta2* dapat diperoleh dari peng-kuadrat-an dikedua sisi persamaan baik Px maupun Py.

$$p_x^2 = l_1^2 c^2\theta_1 + l_2^2 c^2\theta_{12} + 2l_1 l_2 c\theta_1 c\theta_{12}$$
$$p_y^2 = l_1^2 s^2\theta_1 + l_2^2 s^2\theta_{12} + 2l_1 l_2 s\theta_1 s\theta_{12}$$
$$p_x^2 + p_y^2 = l_1^2 (c^2\theta_1 + s^2\theta_1) + l_2^2 (c^2\theta_{12} + s^2\theta_{12}) + 2l_1 l_2 (c\theta_1 c\theta_{12} + s\theta_1 s\theta_{12})$$

3. berdasarkan persamaan $cos^2\theta + sin^2\theta = 1$

$$p_x^2 + p_y^2 = l_1^2 + l_2^2 + 2l_1l_2(c\theta_1[c\theta_1c\theta_2 - s\theta_1s\theta_2] + s\theta_1[s\theta_1c\theta_2 + c\theta_1s\theta_2])$$
$$p_x^2 + p_y^2 = l_1^2 + l_2^2 + 2l_1l_2(c^2\theta_1c\theta_2 - c\theta_1s\theta_1s\theta_2 + s^2\theta_1c\theta_2 + c\theta_1s\theta_1s\theta_2)$$
$$p_x^2 + p_y^2 = l_1^2 + l_2^2 + 2l_1l_2(c\theta_2[c^2\theta_1 + s^2\theta_1])$$
$$p_x^2 + p_y^2 = l_1^2 + l_2^2 + 2l_1l_2c\theta_2$$

4. maka di dapatkan :

$$\cos\theta_2 = \frac{p_x^2 + p_y^2 - l_1^2 - l_2^2}{2l_1l_2}$$

5. Berdasarkan $cos^2\theta + sin^2\theta = 1$

$$\sin\theta_2 = \pm\sqrt{1 - \left(\frac{p_x^2 + p_y^2 - l_1^2 - l_2^2}{2l_1l_2}\right)^2}$$

Maka dari cos dan sin teta2 dapat diperoleh teta2-nya yaitu

$$\theta_2 = A\tan 2\left(\pm\sqrt{1 - \left(\frac{p_x^2 + p_y^2 - l_1^2 - l_2^2}{2l_1l_2}\right)^2}, \frac{p_x^2 + p_y^2 - l_1^2 - l_2^2}{2l_1l_2}\right)$$

6. untuk mencari *Theta1* dengan persamaan *Cosθ* dan *Sinθ* didapatkan solusi sebagai berikut

$$c\theta_1p_x = l_1c^2\theta_1 + l_2c^2\theta_1c\theta_2 - l_2c\theta_1s\theta_1s\theta_2$$
$$s\theta_1p_y = l_1s^2\theta_1 + l_2s^2\theta_1c\theta_2 + l_2s\theta_1c\theta_1s\theta_2$$
$$c\theta_1p_x + s\theta_1p_y = l_1(c^2\theta_1 + s^2\theta_1) + l_2c\theta_2(c^2\theta_1 + s^2\theta_1)$$

7. Persamaan sederhana yang diperoleh sebagai berikut

$$c\theta_1p_x + s\theta_1p_y = l_1 + l_2c\theta_2$$

8. untuk mencari Theta1 dengan persamaan - *Sinθ* dan *Cosθ* didapatkan solusi sebagai berikut

$$-s\theta_1p_x = -l_1s\theta_1c\theta_1 - l_2s\theta_1c\theta_1c\theta_2 + l_2s^2\theta_1s\theta_2$$
$$c\theta_1p_y = l_1s\theta_1c\theta_1 + l_2c\theta_1s\theta_1c\theta_2 + l_2c^2\theta_1s\theta_2$$
$$-s\theta_1p_x + c\theta_1p_y = l_2s\theta_2(c^2\theta_1 + s^2\theta_1)$$

9. Persamaan sederhana yang diperoleh sebagai berikut

$$-s\theta_1p_x + c\theta_1p_y = l_2s\theta_2$$

10. *Cosθ* diperoleh dari sisi persamaan baik Px maupun Py.

$$c\theta_1 p_x^2 + s\theta_1 p_x p_y = p_x(l_1 + l_2 c\theta_2)$$
$$-s\theta_1 p_x p_y + c\theta_1 p_y^2 = p_y l_2 s\theta_2$$
$$c\theta_1(p_x^2 + p_y^2) = p_x(l_1 + l_2 c\theta_2) + p_y l_2 s\theta_2$$

11. Hasil yang didapatkan

$$\cos\theta_1 = \frac{p_x(l_1 + l_2 c\theta_2) + p_y l_2 s\theta_2}{p_x^2 + p_y^2}$$

12. *sinθ* yang diperoleh dengan persamaan sebagai berikut

$$\sin\theta_1 = \pm\sqrt{1 - \left(\frac{p_x(l_1 + l_2 c\theta_2) + p_y l_2 s\theta_2}{p_x^2 + p_y^2}\right)^2}$$

13. Sebagai hasilnya, kemungkinan solusi untuk *Theta1* dapat ditulis

$$\theta_1 = A\tan 2\left(\pm\sqrt{1 - \left(\frac{p_x(l_1 + l_2 c\theta_2) + p_y l_2 s\theta_2}{p_x^2 + p_y^2}\right)^2}, \frac{p_x(l_1 + l_2 c\theta_2) + p_y l_2 s\theta_2}{p_x^2 + p_y^2}\right)$$

2.4. Sistem Kendali Lintasan Gerak *Robot manipulator*

Forward dan *inverse kinematics* merupakan **s**olusi dari kontrol kinematika *Robot manipulator*, yang hanya memperhatikan aspek geometri dari robot. Solusi ini tidak memperhatikan batasan-batasan lain yang diberlakukan oleh *workspace* di mana robot beroperasi. Secara khusus, *forward* dan *inverse kinematics* tidak memperhitungkan kemungkinan akan tabrakan antara *Robot manipulator* dengan benda di sekitar *workspace.* Oleh karena itu dibutuhkan sebuah sistem perencanaan lintasan gerak robot (*path planning*), sehingga robot akan mampu bergerak pada titik-titik lintasan (path points) yang meliputi titik awal, titik-titik singgah hingga titik tujuan

Tujuan pokok gerak perencanaan adalah untuk menghasilkan fungsi yang robot akan bergerak. Fungsi yang akan dihasilkan tergantung pada jenis pekerjaan robot atau tidak manipulator. Sebagai contoh, point-to-point bergerak seperti operasi yang memilih dan tempat, hanya titik awal dan akhir harus ditetapkan. Jalan terus-menerus bergerak, seperti pengelasan, sejumlah intermediate poin selain titik awal dan akhir perlu ditentukan juga. Fungsi yang mengatur gerakan robot yang

dihasilkan oleh gerak controller. Masukan untuk gerak controller ditentukan oleh algoritma generator lintasan. Algoritma ini dirancang berdasarkan parameter yang diperlukan untuk menggambarkan jenis gerakan (point-to-point atau kontinu) robot perlu menjalani. Juga memperhitungkan aspek yang penting

Lintasan dapat direncanakan di space bersama atau di ruang Cartesian. Dalam sendi-ruang perencanaan, masukan untuk gerak controller ditentukan dalam hal posisi sendi, kecepatan dan percepatan. Dalam perencanaan ruang Cartesian, masukan yang akhir-efektor posisi, orientasi dan turunannya waktu. Tujuan utama dari perencanaan untuk robot kami akan mengeksekusi lintasan ditentukan dengan kecepatan yang berbeda-beda sebagai manipulator bergerak sepanjang itu. Setiap sendi pada robot yang biasanya DC servo motor, dapat dikontrol secara mandiri. Jadi, untuk tujuan kami kami telah memutuskan untuk pergi untuk terus-menerus-path perencanaan

Ada berbagai cara dalam melakukan path planning, dimana semuanya memiliki tujuan yang sama yaitu dengan memberikan serangkaian titik yang disebut via points pada sepanjang lintasan. Cara yang paling sederhana dalam melakukan path planningadalah dengan memberikan serangkaian urutan posisi end-effector. Pada cara ini inverse kinematics dibutuhkan dalam mengubah posisi *end-effector* menjadi konfigurasi tiap sendi

Agar robot dapat bergerak dengan tepat ke setiap path points hasil path planning tadi, maka diperlukan perencanaan gerak. Perencanaan gerak ini dilakukan dengan membuat fungsi posisi berdasarkan waktu. Fungsi waktu tersebut biasa dengan sebutan tracjectory. Karena tracjectory merupakan fungsi waktu, maka dapat dihitung juga besar kecepatan dan percepatan robot sepanjang jalur lintasan. Perencanaan trajectory atau trajectory planning dapat dilakukan baik dengan joint space maupun cartesian space.

2.4.1. *Path planning* dengan *interpolasi linear*

Untuk mengikuti jalur lintasan yang telah ditentukan, maka yang terlebih dahulu dilakukan adalah dengan menentukan titik-titik singgah (via points) pada lintasan atau biasa disebut path planning. Ada banyak algoritma path planning, salah satunya adalah dengan menggunakan interpolasi linier

Persamaan umum untuk interpolasi linier adalah sebagai berikut,

$$P(t) = P0 + y = \frac{f(P)}{k}.t$$

Dimana :

f(P) = fungsi lintasan

Po = koordinat awal

k = konstanta (waktu untuk menempuh jarak dari koordinat awal hingga akhir)

t = time sampling

Selanjutnya dari persamaan interpolasi linier tersebut diturunkan ke dalam jalur lintasan yang diinginkan. Lintasan tersebut dapat berupa garis lurus, lingkaran.

Untuk lintasan garis lurus dengan koordinat awal P1(x1,y1) dan koordinat akhir P2(x2,y2,z2) didapat persamaan berikut

$$Px(t) = x1 + \frac{x2 - x1}{k}.t \qquad Py(t) = y1 + \frac{y2 - y1}{k}.t$$

Sedangkan untuk lintasan berupa garis lingkaran dengan pusat koordinat di P(x,y) dan jari – jari r, maka didapat persamaan berikut,

$$Px = \mathrm{x0} + \mathrm{R}.\cos\left(\frac{2\pi.\mathrm{t}}{k}\right) \qquad Py = \mathrm{y0} + \mathrm{R}.\sin\left(\frac{2\pi.\mathrm{t}}{k}\right)$$

Perencanaan trajectory dengan secara langsung menentukan konfigurasi atau sudut masing-masing sendi dari titik awal hingga titik akhir pergerakan robot disebut dengan joint space trajectory planning. Perencanaan pada *joint space* lebih sederhana dan cepat, dikarenakan tidak diperlukan path planning dan perhitungan inverse kinematics pada setiap perpindahan path point. Denganpemanfaatan perencanaan ini, maka titik akhir dapat dicapai setepat mungkin sesuai dengan acuan yang diberikan. Hal ini disebabkan kemampuannya dalam mengatasi batasan (*constrain*) dari sistem. Contohnya adalah jika pergerakan sendi harus dimulai dan berhenti dengan kecepatan nol. Perencanaan ini banyak diaplikasikan pada robot atau mesin dengan orientasi gerak titik ke titik (*point to point motion*) seperti mesin bor. Namun dalam aplikasinya tersebut biasanya ditambahkan perhitungan *inverse kinematics* diawal untuk mengkonversi koordinat titik awal dan titik akhir end effector menjadi konfigurasi sudut awal dan akhir sendi – sendinya. Kerugian dari trajectory planning ini adalah bahwa posisi *end effector* tidak secara langsung terkontrol dan tidak dapat menghindari halang rintang pada *workspace*

2.4.3. *Cartesian Trajectory planning*

Trajectory planning yang terbaik dilakukan dengan menentukan secara pasti posisi dan orientasi *end-effectror* pada semua *path points*. Perencanaan trajectory seperti ini dikenal dengan *cartesian trajectory planning*. Untuk mencapai *trajectory planning* ini, maka pada setiap waktu sampling sistem harus memperoleh data sudut setiap sendi lalu melakukan konversi ke posisi dan orientasi *end-effector* menggunakan *forward kinematics*. Setelah itu dihitung nilai kesalahan *trajectory* yang terjadi, dan akan ditentukan posisi berikutnya. Posisi yang didapat tersebut akan dikonversi lagi menjadi sudut-sudut sendi dengan *inverse kinematics* untuk kemudian dieksekusi menjadi perpindahan posisi. Dengan *trajectory planning* seperti ini maka secara langsung posisi *end-effector* dapat selalu dipantau dan dikoreksi.

Cartesian trajectory planning baik digunakan pada *Robot manipulator* yang dalam fungsinya mengharuskan *end-effector* untuk mengikuti jalur lintasan tertentu *(comtinuos path tracking)* seperti mesin pengecatan. Namun karena waktu komputasinya yang cukup lama membuat *trajectory planning* ini jarang dipilih. Pada aplikasi yang sebenarnya digunakan gabungan antara cartesian dan *joint space trajectory planning*, dimana posisi *end-effector* hanya dikendalikan pada setiap titik-

titik singgah end-effector. Sistem akan mengambil data sudut setiap sendi pada titik singgah, dan mengganti sudut acuan setiap sendi pada titik singgah tersebut dengan data yang didapat. Sudut acuan setiap sendi pada titik singgah tersebut sebelumnya didapat dari perhitungan *path points* di awal pergerakan robot. Dengan pendekatan ini diharapkan sistem sudah bisa mengikuti lintasan yang diinginkan

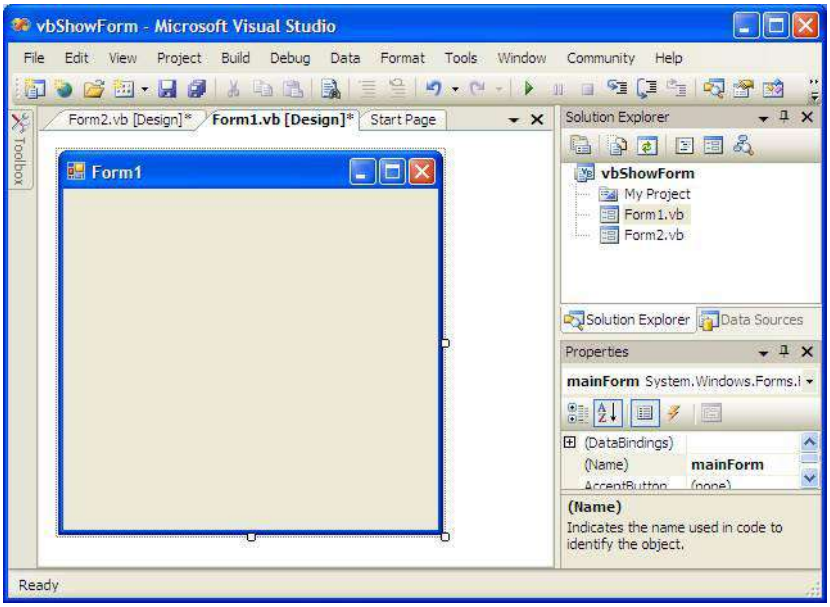
2.5 *Software Berbasis .NET*

Dalam aplikasi industri, *Robot manipulator* adalah teknologi otomasi yang fleksibel dimana tidak seperti pada otomasi tetap yang merupakan sebuah mesin yang khusus hanya dapat mengangani suatu pekerjaan spesifik tertentu. Kefleksibilitasan tersebut dapat dicapai karena sebuah *Robot manipulator* dapat diprogram ulang atau pun dapat dikendalikan secara remote control oleh operator melalui komputer.

Dalam melakukan program ulang ataupun kendali remote control, dibutuhkan sebuah program antarmuka pada komputer. Pembuatan program antarmuka tersebut dapat menggunakan banyak sekali bahasa pemrograman. Salah satunya adalah C# (baca : See-Sharp) programming language yang merupakan salah satu bahasa pemrograman baru dan menjanjikan banyak kemudahan bagi programer

C# adalah bahasa pemrograman baru yang diciptakan Microsoft yang digunakan oleh banyak developer .NET untuk mengembangkan aplikasi dengan platform .NET. .NET sendiri merupakan suatu komponen tambahan Windows yang terintegrasi yang dibuat dengan tujuan pengembangan berbagai macam aplikasi. C# bersifat sederhana karena dibuat berdasarkan bahasa C/C++ dan sudah mendukung Object Oriented Programming

Pada pemrograman *Visual*, pengembangan aplikasi dimulai dengan pembentukan *user interface*, kemudian mengatur properti dari objek-objek yang digunakan dalam *user interface*, dan baru dilakukan penulisan kode program untuk menangani kejadian-kejadian (*event*). Tahap pengembangan aplikasi demikian dikenal dengan istilah pengembangan aplikasi dengan pendekatan *Bottom Up.*



Gambar 2. 10 *Visual Studio .Net*

Ada beberapa hal yang harus dipahami dalam mempelajari Visual Studio :

1. Objek

   Sering disebut *entity* adalah sesuatu yang bisa dibedakan dengan lainnya. Pada dasarnya seluruh benda didunia bisa dikatakan sebagai objek, contoh : mobil, komputer, radio, dan lain-lain.

   Dalam *Visual studio* objek-objek yang dimaksud disebut kontrol. Jenis-jenis kontrol antara lain *: Label, Text Box, Combo Box, List Box,* dan masih banyak lagi.

2. Properti

   Sering disebut atribut, adalah ciri-ciri yang menggambarkan suatu objek. Misalnya disebut objek mobil jika mempunyai ban, spion, rem, dan lain-lain.

3. *Event*

   Suatu kejadian yang menimpa objek. Bagaimana jika mobil didorong, ditabrak, dicat dan sebagainya.

4. Metode

   Kemampuan yang dimiliki oleh suatu objek. Contohnya jika mobil berbelok, mundur, maju**.**

2.6 Sistem Kontrol

2.6.1 Stepper motor

Motor stepper adalah suatu alat penggerak yang memanfaatkan gaya tarik magnet. Rotornya berhenti pada posisi kutub yang dieksitasi oleh arus yang mengalir pada lilitan. Rotor pada motor biasanya berputar secara kontinyu jika motor dieksitasi, tetapi rotor pada *motor stepper* berubah dari posisi diam dengan mengubah eksitasi kutub.

Arus yang mengalir pada setiap lilitan hanya sesaat sehingga bentuk arusnya berupa pulsa. *Rotor* berputar karena pulsa yang bergantian. Kecepatan putaran *rotor* ditentukan oleh kecepatan perpindahan pulsa dan sudut putaran sebanding dengan banyaknya pulsa yang diberikan. Apabila satu pulsa input menghasilkan perputaran sejauh 1,8 derajat, sehingga 20 pulsa akan menghasilkan perputaran penuh sebesar 36 derajat dan untuk mendapatkan satu putaran penuh 360 derajat dibutuhkan 200 pulsa.

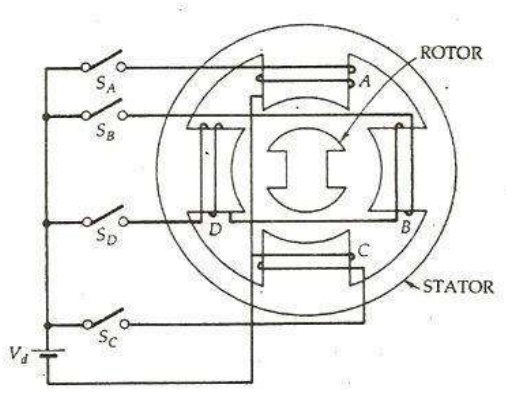
Rotor yang digunakan terbuat dari baja lunak dan memiliki sejumlah gigi yang jumlahnya kurang dari jumlah kutub pada stator. *Stator* memiliki beberapa pasang kutub dimana setiap pasang kutub diaktifkan melalui prinsip elektromagnetik oleh arus yang mengalir melalui kumparan yang dililitkan pada masing – masing kutub. Pada saat sepasang kutub dalam keadaan aktif sehingga akan timbul medan magnet yang kemudian menarik pasangan gigi *rotor* terdekat, sehingga gigi akan bergerak ke posisi segaris dengan kutub.

Untuk menggerakkan sebuah *motor stepper* setiap pasang kumparan stator harus disambungkan dengan aliran listrik dan diputuskan secara bergantian dalam urutan yang benar. Dengan demikian, input ke motor berupa deretan pulsa yang menghasilkan output ke setiap pasang kumparan *stator*.

Sistem penggerak yang biasa digunakan terdiri dari dua blok utama yaitu pengatur urutan logika dimana menerima pulsa – pulsa input dan menghasilkan pulsa – pulsa output dalam urutan sebagai mana yang dibutuhkan untuk mengontrol penggerak agar menghasilkan pulsa output dengan amplitudo yang sesuai.

Motor langkah (*stepper*) banyak digunakan dalam berbagai aplikasi, dipergunakan apabila dikehendaki jumlah putaran yang tepat atau diperlukan sebagian dari putaran motor.

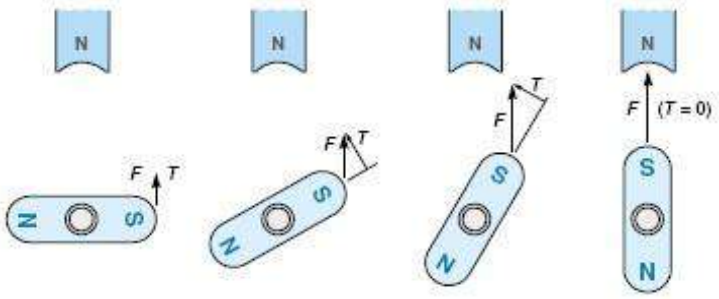
Aplikasi penggunaan motor langkah dapat juga dijumpai dalam bidang industri atau untuk jenis motor langkah kecil dapat di gunakan dalam perancangan suatu alat mekatronik atau robot. Pada gambar 2.1 berikut ditunjukkan dasar susunan sebuah motor langkah (*stepper*).



Gambar 2.11 Diagram motor langkah (*stepper*)

2.6.2 Mode Operasi

Motor stepper mempunyai dua mode operasi yaitu ***single step mode*** dan ***slew***



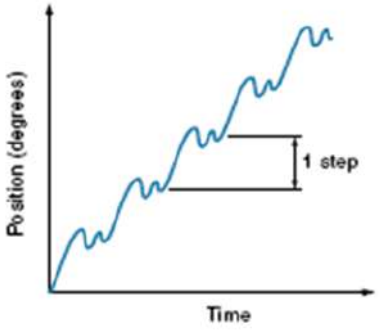
Gambar 2.12 Torsi Rotor Sejajar dengan Kutub medan

mode. Pada *single step mode* atau disebut juga *bidirectional mode,* frekuensi *step* cukup lambat untuk memperbolehkan rotor (hampir semua) berhenti di antara *step*. Gambar 3 menunjukkan sebuah grafik posisi *versus* waktu untuk operasi single *step*. Pada setiap *step*, motor meneruskan sudut tertentu dan kemudian berhenti. Jika motor bebannya kecil, *overshoot* (lonjakan) dan osilasi dapat terjadi pada akhir setiap *step* seperti yang ditunjukkan pada gambar.

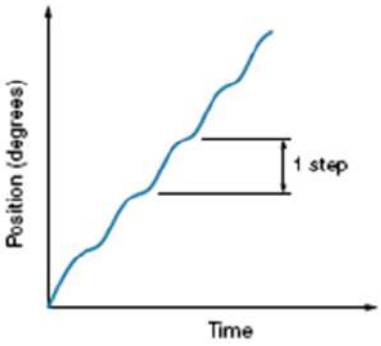
Pada ***slew mode***, atau *unidirectional mode*, frekuensi *step* adalah cukup tinggi sehingga tidak mempunyai waktu untuk berhenti. Mode ini mirip dengan motor listrik biasa (*regular electric motor*). Jadi motor selalu mengalami torsi dan berotasi lebih halus dengan kontinyu. Gambar 4 menunjukkan grafik posisi versus waktu untuk *slew mode*. Walaupun setiap *step* dapat tetap dilihat, gerakannya jauh lebih halus dibandingkan dengan *single-step mode*.

Motor stepper dengan *slew mode* tidak dapat berhenti atau berbalik arah secara mendadak (*instantaneously*). Jika dicoba dilakukan, maka kemungkinan besar *rotational inertia* motor akan membawa rotor ke depan beberapa *step* sebelum berhenti. Jadi *step-count* keseluruhan akan hilang. Kemungkinan untuk menjaga *step-count* dalam *slew mode* dilakukan dengan memperlambat kecepatan lereng- atas dari *single-step mode* dan kemudian pada lereng-bawah bagian akhir dari *slew*. Hal ini berarti kontroler harus mengetahui waktu di depan seberapa jauh motor harus

jalan. Secara tipikal *slew mode* digunakan untuk memperoleh posisi motor dalam "*ballpark*", dan kemudian *fine adjusment* dapat dilakukan dengan *single step*. *Slewing* menggerakkan motor lebih cepat tetapi memperbesar perubahan kehilangan *step-count*.



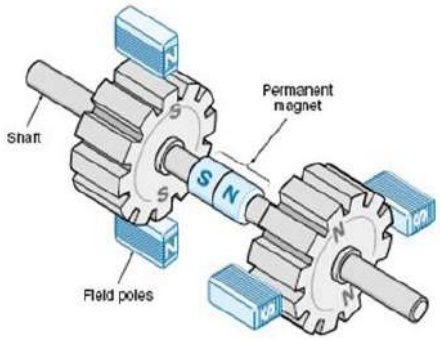
Gambar 2.13 Posisi versus waktu untuk *single-step mode*



Gambar 2.14 Grafik posisi versus waktu untuk *slew mode*

2.6.3 Motor Stepper Hybrid

Motor stepper hybrid menggabungkan kelebihan/fitur motor stepper magnet permanen dan motor stepper *variable reluctance* (VR) dan ini yang paling banyak digunakan saat ini. Rotor bergerigi, yang membolehkan sudut *step* yang sangat kecil ($1{,}8^0$ tipikal), dan mempunyai suatu magnet permanen yang memberikan *detent torque* yang kecil bahkan ketika catu daya dimatikan.



Gambar 2.15 Konstruksi internal motor stepper hybrid (hanya ditampilkan dua kutub per stator)

internal dari motor hybrid yang dapat dianggap lebih rumit dari motor magnet permanen biasa/sederhana. Rotor terdiri dari dua roda bergigi dengan suatu magnet di antaranya—satu roda termagnetisasi secara sempurna menjadi utara dan yang lainnya sempurna menjadi selatan. Untuk setiap *step*, dua gigi berlawanan pada roda utara ditarik menuju dua kutub medan selatan, dan dua gigi berlawanan pada roda selatan ditarik menuju dua kutub medan utara. Kumparan atau perkawatan internal lebih rumit dari motor magnet permanen atau motor VR, tetapi untuk ke dunia luar motor ini sederhana dan mudah untuk dikontrol.

Teori operasi motor hybrid mirip dengan motor VR di mana rotor dan stator mempunyai jumlah gigi yang berbeda dan untuk setiap *step*, gigi yang mendapat energi terdekat yang akan ditarik untuk disejajarkan. Namun, prinsip-prinsip

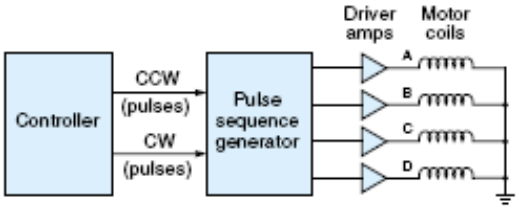
magnetik diperlukan, pada satu waktu kapan saja, setengah kutub-kutub menjadi utara dan setengah lainnya menjadi selatan. Untuk mempertahankan keseimbangan magnetik (magnetic balance), setiap kutub harus dapa men- switch polaritas supaya dapat memberikan kutub yang tepat pada waktu yang tepat. Hal ini diselesaikan dengan satu cara dari dua cara: Untuk motor bipolar, tegangan yang digunakan harus dibalik oleh rangkaian *driver* (seperti pada motor stepper magnet permanen dua-phase). Pada sisi lain, motor unipolar mempunyai dua kumparan terpisah arah berlwanan pada setiap kutub medan (disebut ***bifilar winding***), dan juga setiap kutub dapat menjadi utara atau selatan. Karena itu motor stepper hybrid unipolar tidak memerlukan rangkaian pembalik polaritas.

2.6.4 **Kontrol Motor Stepper**

Untuk menghubungkan motor stepper dengan piranti digital atau *I/O port* dibutuhkan rangkaian *interface*. Hal ini sangat penting karena jumlah arus yang diperlukan untuk memberikan energi (*energizing*) pada pasangan-pasangan kumparan lebih besar dari kemampuan *I/ O port*, sehingga dibutuhkan sejumlah rangkaian penyangga (*buffer*) yang akan menguatkan arus untuk dapat menggerakkan motor stepper.

Rangkaian penggerak (driver) motor stepper

Kontroler yang menentukan jumlah dan arah *step* yang akan diberikan (tergantung aplikasi). *Driver amplifier* memperbesar daya dari sinyal kemudi kumparan. Di sini tidak diperlukan rangkaian pengubah digital ke analog karena kutub-kutub medan adalah on atau off, *driver amplifier* efisien kelas C dapat digunakan.

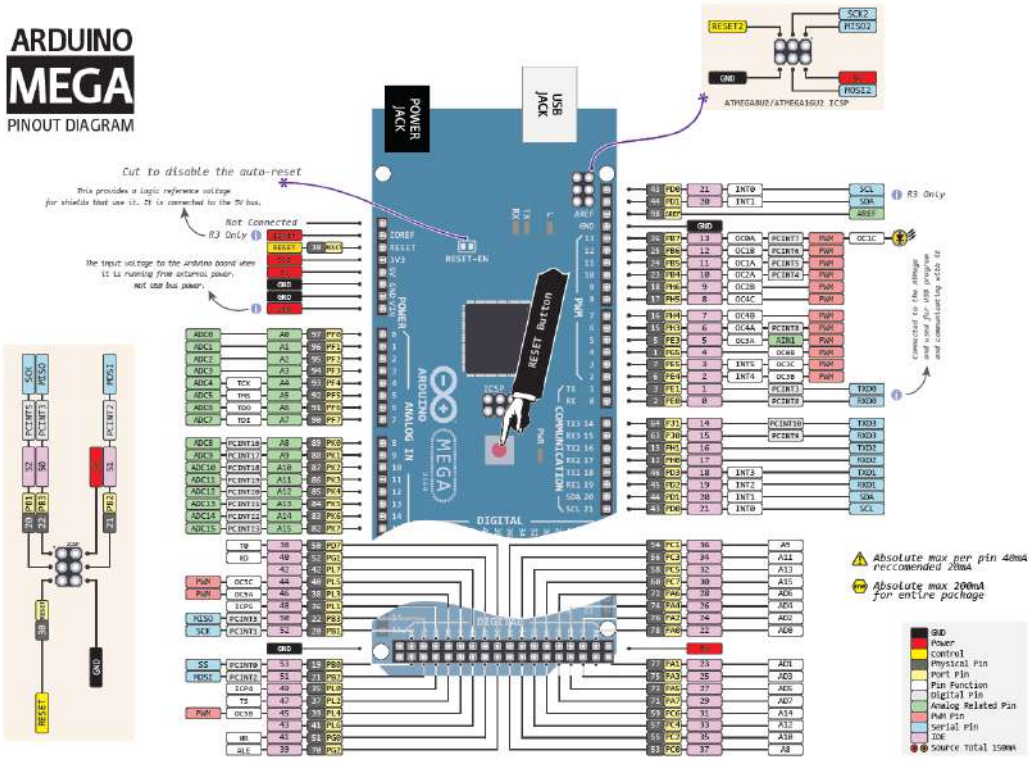


Gambar 2.16 Diagram blok rangkaian kontrol motor stepper

2.6.5 Arduino Interface

Arduino juga menyederhanakan proses bekerja dengan mikrokontroler dan beberapa kelebihannya yaitu tidak perlu perangkat chip programmer karena didalamnya sudah ada bootloadder yang akan menangani upload program dari komputer. Sudah memiliki sarana komunikasi USB, Sehingga pengguna laptop yang tidak memiliki port serial/RS323 bisa menggunakannya. Memiliki modul siap pakai (Shield) yang bisa ditancapkan pada board arduino. Contohnya shield GPS, Ethernet, dan lain sebagainya. Arduino mega 2560 adalah papan mikrokontroler berdasarkan ATmega2560 (*datasheet*). Ini memiliki 54 digital pin input / output (pin 15 dapat digunakan sebagai output PWM (*Pulse Width Modulation*)) , 16 analog input , 4 UART (*Universal Asyncronous Receiver Transmitter*) (hardware port serial) , osilator kristal 16 MHz , koneksi USB , jack listrik , header ICSP (*In-Circuit Serial Programming*) , dan tombol reset. Semuanya diperlukan untuk mendukung kerja mikrokontroler, cara mengaktifkan Arduino mega 2560 adalah dengan menghubungkannya ke komputer dengan kabel USB atau memberikan dengan adaptor AC - DC atau baterai. Arduino Mega ini *compatible* dengan Arduino Duemilanove atau Diecimila.

Mega 2560 adalah update dari Arduino Mega. Mega 2560 berbeda dari semua *board* sebelumnya yang tidak menggunakan FTDI (*Future Technology Devices International*) chip driver USB - to -serial. Revisi ke 2 dari *board* Mega 2560 memiliki resistor 8U2, sehingga lebih mudah untuk dimasukkan ke dalam mode DFU (*Device Firmware Update*).



Gambar 2. 17 Pin Configuration Arduino Mega

Dibawah ini spesifikasi dari arduino mega:

Tabel 2.2 Spesifikasi arduino Mega 2560

Mikrokontroler	Atmega2560
Tegangan Operasi	5V
Input Voltage (disarankan)	7-12V
Input Voltage (limit)	6-20V
Pin Digital I/O	54(15 pin digunakan sebagai output PWM)

Pins Input Analog	16
Arus DC per pin I/O	40 mA
Flash Memory	256KB (Atmega 2560) atau 8 KB digunakan oleh *Bootloader*
SRAM	8 KB
EEPROM	4 Kb
Clock Speed	16 MHz
Ukuran	1.85cm x 4.3cm

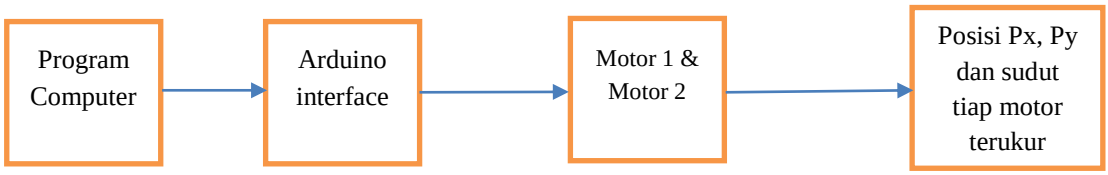
BAB III

PERANCANGAN ALAT DAN SIMULASI

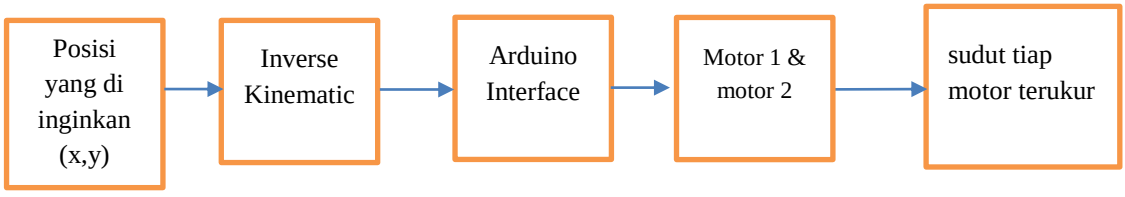
Perancangan ini meliputi pembuatan perangkat keras dan perangkat lunak, Perancangan perangkat lunak yang dimaksud adalah pembuatan program pada dengan *software visual studio*. Program yang dirancang meliputi program *invers kinematic* untuk gerak garis lurus dan lingkaran

3.1. Prinsip Kerja *Robot manipulator*

Robot Manipulator memilki 2 DOF (*degree of freedom*) dan *end-effector*. Dalam penelitian ini menerapkan metode *inverse kinematic* dalam melakukan pengujian. Blok diagram perancangan perangkat sistem di tunjukkan pada gambar 3.1. Sedangkan blok diagram perancangan perangkat lunak atau software di tunjukkan pada gambar 3.2



Gambar 3.1. Blok Diagram Sistem

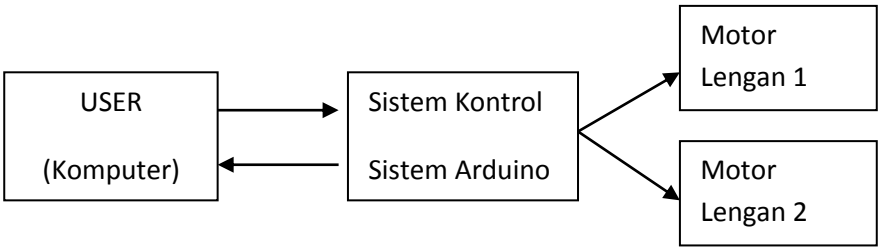


Gambar 3.2. Blok Diagram Software

Pada blok diagram perangkat lunak, menggunakan metode *inverse kinematic*. *Arduino* memiliki fungsi sebagai *interface* untuk mengatur posisi *motor stepper* sebagai aktuator penggerak *end-effector*. Posisi *end-effector* di gunakan untuk menunjukan posisi koordinat akhir posisi X dan Y.

3.2. Perancangan System *Robot manipulator*

Robot manipulator dapat dikendalikan secara langsung oleh pengguna menggunakan komputer dan data yang dikirim oleh komputer menuju ke sistem Arduino yang berfungsi untuk memproses dan mengolah data dari program untuk mengirimkan perintah untuk menggerakan *motor stepper*. *Motor stepper* berfungsi sebagai aktuator untuk menggerakan tiap sendi *Robot manipulator*. Jalur komunikasi serial digunakan penghubung antara sistem Arduino dan komputer.



Gambar 3.3 Blok Diagram system Pengendali *Robot manipulator*

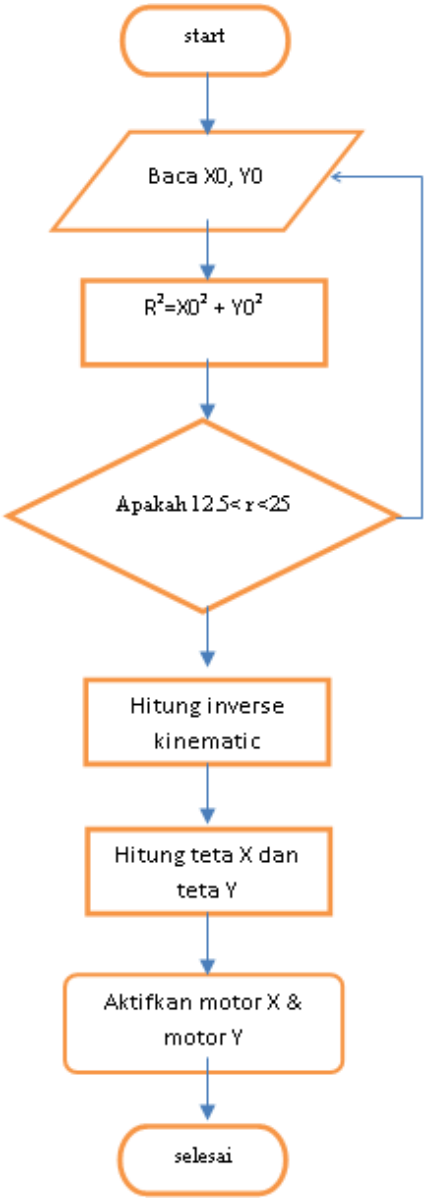
3.3 Perancangan Progam Visual Studio

Perancangan Perangkat Lunak Pada penelitian ini, dirancang sebuah aplikasi untuk mengendalikan *Robot manipulator* menggunakan perangkat lunak *Software Visual studio 2012* yang berfungsi sebagai antarmuka. Pada form aplikasi sistem pengendali *Robot manipulator* terdapat beberapa button yang memiliki fungsi, tampilan aplikasi sistem pengendali *Robot manipulator*. Pada aplikasi sistem pengendali *Robot manipulator*, data di input berupa Koordinat X dan Y dan sudut pada tiap sendi. Data ini akan dikirim ke *arduino* sebagai *interface*. Data yang dikirim berupa angka dan huruf. Dimana huruf berfungsi untuk mengaktifkan Notasi koordinat X dan Y, Sebelum dikirim ke *Arduino* data akan mengalami pengabungan data. Kemudian data yang dikirim ke *Arduino* akan mengalami proses parsing data. Parsing data adalah suatu cara memecahkan data yang berupa masukan dari keyboard..

Sebuah GUI (*graphical user interface*) memudahkan pengguna untuk berinteraksi secara visual dengan program. GUI memberikan program penampilan (atau LAF, *look-and-feel*) yang menarik. Penyediaan komponen-komponen GUI akan memudahkan pengguna untuk memprogram lebih cepat.

GUI dibangun dari kendali-kendali GUI (yang kadang kala dikenal dengan komponen atau *widget*, singkatan dari *window gadget*). Setiap kendali GUI merupakan objek yang dapat menampilkan informasi pada layar atau yang memampukan pengguna untuk berinterasi dengan sebuah aplikasi melalui *mouse*, papanketik (*keyboard*), atau masukan lainnya (seperti *voice command*).

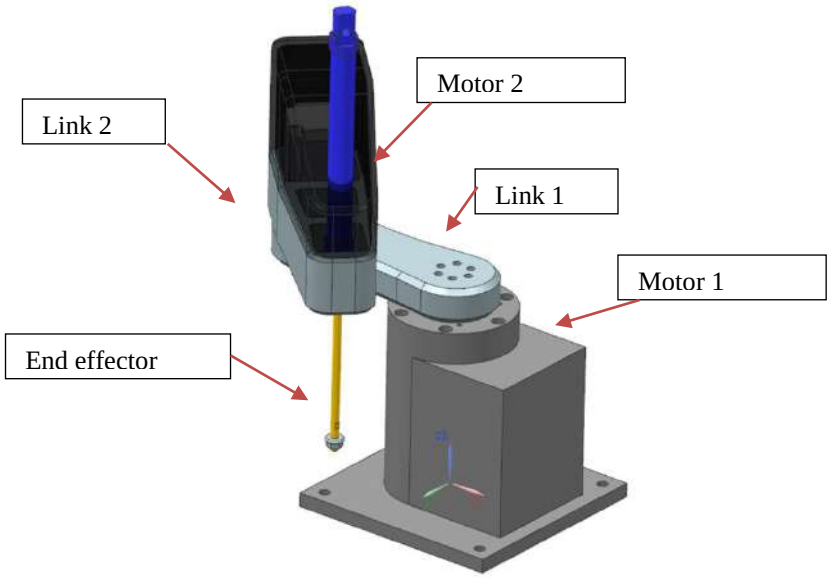
Berikut *flowchart* yang di gunakan dalam pembuatan GUI pada penelitian ini yang mana dengan dilakukan dengan identifikasi koordinat x dan y yang kemudian akan di terjemahkan oleh system pada aplikasi GUI pada *visual studio* dengan dasar data atau informasi mengenai robot. Dari data data tersebut akan di hirung dengan mengunakan rumus *Inverse kinematic* untuk menghasilkan nilai sudut yang akan dikirim melalui komunikasi serial dengan *arduino interface* menjadi sebuah gerakan motor pada joint lengan 1 dan joint lengan 2



Gambar 3.4. *flowchart inverse kinematic*

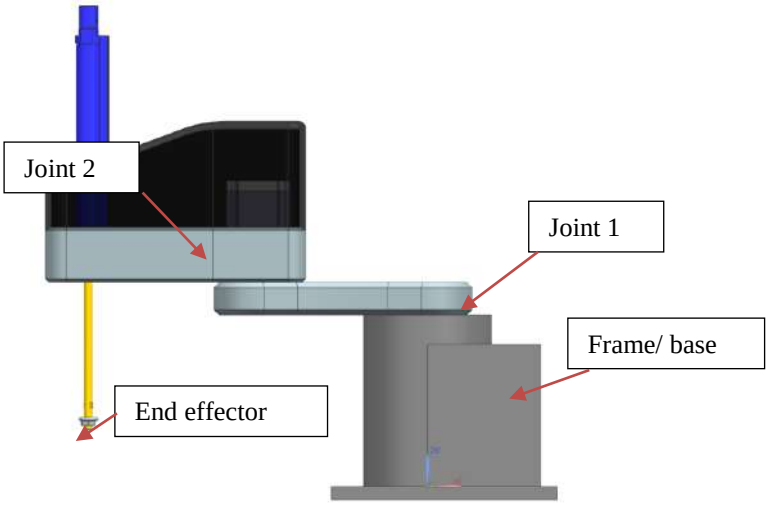
3.4 *Frame Robot Manipulator*

Pada *Robot Manipulator* memiliki 2 DOF *(degree of fredom)* derajat kebebasan dan *pen drawing (end effector).* Bagian-bagian pada robot lengan terbuat dari dan logam alumunium yang berfungsi sebagai *link*, 2 buah motor stepper yang berfungsi sebagai aktuator dan sebuah *pen drawing* yang berfungsi sebagai *end efector*. Pengendalian lengan dengan *interface Arduino* dan Komputer sebagai program simulasinya. Gambar 3.5 menunjukkan struktur mekanik robot lengan



Gambar 3.5. Struktur mekanik *Robot Manipulator*

Setiap *link* diberi nomor dimulai dari bagian *base* yang tidak bergerak. *Frame (motor 1).* Batang yang bergerak dan berhubungan dengan *frame* disebut *link* 1. lengan yang bergerak dan berhubungan dengan *link* 1 disebut *link* 2 dan . Antara *link1* dengan *link2* di hubungkan dengan *joint*. *Joint1* dan *joint2* adalah *joint* yang menghubungkan *link1* dengan *link2*. *Frame* merupakan sistem koordinat yang menggambarkan posisi sebuah *link* relatif terhadap *link* lainnya. Sistem koordinat ini melekat pada *link*. Penomoran *frame* pada *Robot Manipulator* ditunjukkan pada gambar 3.6.



Gambar 3.6. link joint *Robot Manipulator*

Workspace	250 mm
Akurasi	
X	±0.015 mm
Y	±0.015 mm
Theta	$\pm 0.005^{0}$
Joint Ranges	
Link 1	$\pm 155^{0}$
Link 2	$\pm 145^{0}$

2.1 Basic specification of Robot body (4 axes)

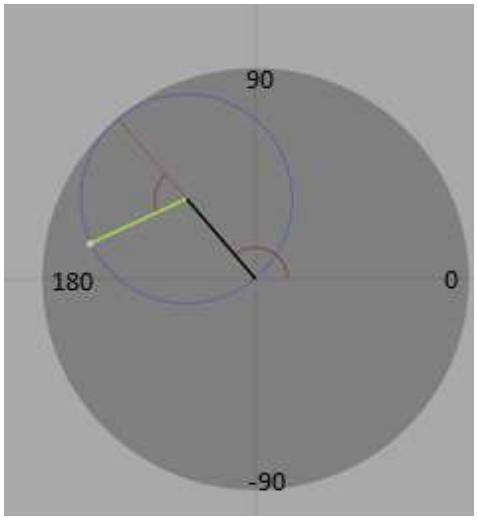
Item		Specifications			
Robot model		EZ03V4-02 -4525	EZ03V4-02 -4515	EZ03F4-02 -5525	EZ03F4-02 -5515
Construction		SCARA			
Number of axis		4			
Drive system		AC servo motor			
Max. motion range	Axis 1	250mm	150mm	250mm	150mm
	Axis 2	±170 °			
	Axis 3	±180 °		±145 °	
	Axis	±360 °			
	Axis 5	---			
	Axis 6	---			
Max. speed *5	Axis 1	1400mm/s	1200 mm/s	1400mm/s	1200mm/s
	Axis 2	450 °/s			
	Axis 3	720 °/s			
	Axis 4	2400 °/s			
	Axis 5	---			
	Axis 6	---			
Max. pay load		2 kg (3 kg)			
Max. allowable moment of inertia of wrist *1	Axis 4	0.05 kg·m^2 (Rated 0.005 kg·m^2)			
	Axis 5	---			
	Axis 6	---			
Position repeatability *2		±0.014mm			
Max. reach		450mm		550mm	
Air piping		φ6×2			
Application signal wires		10 wires			
Installation		Ceiling setting		Floor setting	
Ambient conditions		Temperature: 0 to 45 °C *3 Humidity: 20 to 85%RH (No dew condensation allowed) Vibration to the installation surface: Not more than 0.5G (4.9 m/s2)			
Dust-proof / Drip-proof performance *4		IP20			
Noise level *6		70 dB			
Robot mass		42kg		43kg	

1[rad] = 180/π[°], 1[N·m] = 1/9.8[kgf·m]
- On controller display, axis 1 to 6 is displayed as J1 to J6 for each.
- The specification and externals described in this specification might change without a previous notice for the improvement
- Explosion resistance is not available.

*1: The Allowable moment of inertia of a wrist changes with load conditions of a wrist. *2: This value conforms to "JIS B 8432".
*3: Permitted height is not higher than 1,000m above sea level. If used in higher place, permitted temperature is affected by height.
* *4: Liquid such as organic compound, acidity, alkalinity, chlorine or gasoline cutting fluid which deteriorates the seal material are not available to use.
*5: Max. speed on the table shown is maximum value, so it changes depending on the work program and condition of wrist load.
*6: A load equivalent noise level, measured according to the JIS Z 8737-1 (ISO 11201). (Operation at rated load and Max. speed)

3.5. *Workspace Robot Manipulator*

Workspace Robot Manipulator adalah total luas yang memungkinkan terlewati atau dapat dijangkau oleh gerakan *Robot Manipulator* . *Workspace* dapat kita tentukan dengan cara menggerakkan tiap motor secara berurutan sehingga kita dapat mengetahui jangkauan *Robot Manipulator* dan keterbatasan fisik robot. Gambar 8 menunjukkan *workspace Robot Manipulator* dengan jangkauan minimum dan maksimum yang dapat di capai.

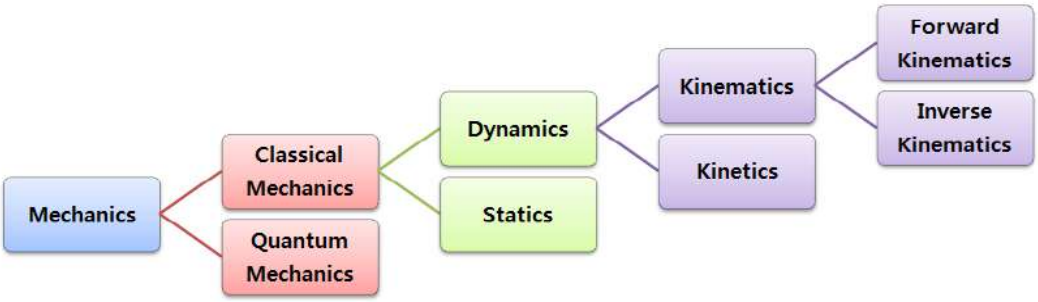


Gambar 3.7. *Workpace robot manipulator*

Motor stepper yang digunakan pada *robot manipulator* dalam pergerakannya mengunakan sudut yaitu dengan dasar penggunaan daerah quadran. Quadran I sudut 0 sampai 90 derajat Quadran II 90 sampai 180 derajat, quadran III 0 sampai -90, quadran IV 0. Pergarakan antara motor 1 dan motor 2 akan terbentuk juring lingkaran dengan jari-jari 25 cm. Kondisikan *motor stepper* bagian *link1* pada kondisi sudut maks atau min lalu gerakan *motor stepper* bagian *link 2* sehingga kita dapatkan ruang gerak baru yang terlewati.

3.6. *Kinematika*

Kinematik merupakan pembelajaran pergerakan robot tanpa memperhitungkan gaya, torsi maupun moment tertentu yang menyebabkan pergerakan. Kinematik yang akan dijelaskan disini adalah kinematik yang khusus mempelajari dan menganalisa pergerakan lengan robot manipulator.



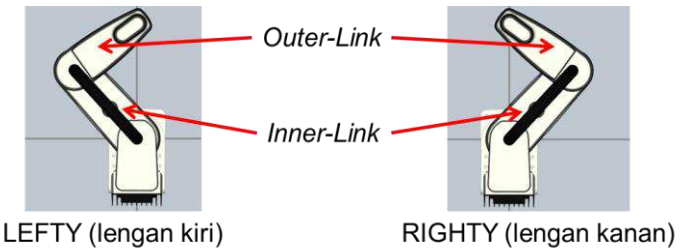
Gambar 3.8. Blok Diagram Kinematika

inverse kinematic digunakan untuk mendapatkan nilai sudut tiap sendi *robot manipulator* dengan memasukan posisi dan orientasi dari *end effector*. Pada *inverse kinematic* menggunakan metode geometris. Gambar 3.7 workpace area manipulator adalah pergerakan untuk sudut , berdasarkan wilayah pergerakan sudut bergerak diwilayah horizontal atau berada pada sumbu x dan sumbu y.

3.6.1 Penyelesaian persamaan *inverse kinematic*

Permasalahan *inverse kinematic* dengan menentukan posisi dari robot dan orientasi dari *end-effector*. Permasalahan dari *inverse kinematic* tersebut mempunyai banyak solusi yang didapat atau disebut juga dengan *multiple solution.*

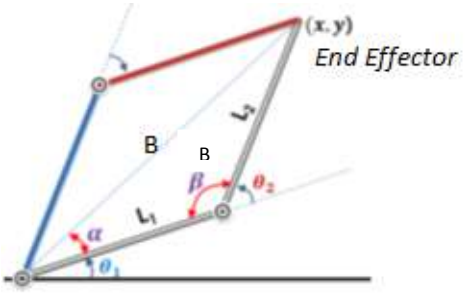
Dalam hal ini permasalahan *inverse kinematic* yang dibahas tentang permasalahan di bagian posisi *robot manipulator*. Pada permasalahan posisi ini ada beberapa yang harus diperhatikan dalam permasalahan *inverse kinematic* ini, yaitu:



Gambar 3.9 gambar pergerakan robot kiri dan kanan

Dalam permasalahan *inverse kinematics* diperhatikan pada bagian posisi, dimana yang di lakukan percobaan pada pagian *inner link, outer link* dan *end effector*

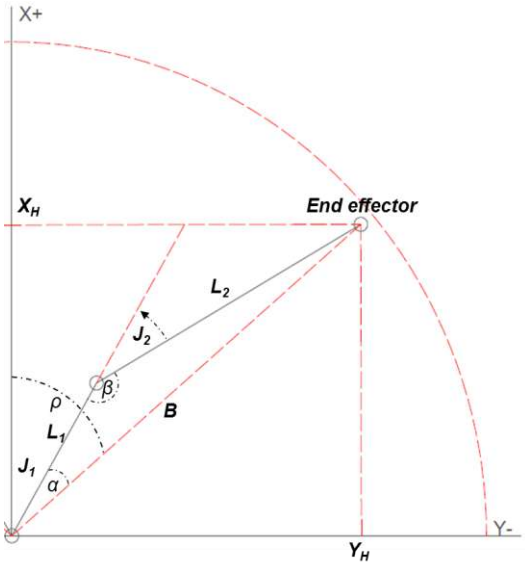
untuk posisi *robot manipulator.* Oleh karena itu, Persamaan *inverse kinematic* dalam menentukan titik posisi robot harus di lihat metode pergerakan yang akan di gunakan



Gambar 3.10 Model gerak *Inverse kinematic* Kiri dan kanan

Langkah Menemukan *Sudut Joint* untuk *Inverse kinematics* Untuk permasalahan *inverse kinematics* menggunakan solusi geometri, maka akan dibuat persamaan pada tiap-tiap sudut joint *robot manipulator*. Solusi pada Joint Pertama Pada permasalahan *inverse kinematics* ini pada joint pertama yang diamati pada pergerakan lengan kiri dan lengan kanan. Hal ini dapat dilihat pada gambar Di atas

3.6.2 Penyelesaian persamaan *Inverse Kinematics* Kiri



- Bagian Kartesian :
  X_h = Posisi kerja dalam sumbu X
  Y_h = Posisi kerja dalam sumbu Y
- Bagian Sudut Servo :
  J_1 = Posisi sudut J1
  J_2 = Posisi sudut J2
- Definisi Tambahan :
  L_1 = Panjang *Inner-Link*
  L_2 = Panjang *Outer-Link*
  B = Diagonal antara X_h dan Y_h
  ρ = Sudut dalam antara X_h dan B
  α = Sudut dalam antara L_1 dan B
  β = Sudut dalam antara L_1 dan L_2

Gambar 3.11 Persamaan *Inverse Kinematics* Kiri

Dengan *theorema Phytagoras*, *trigonometri*, dan Cosinus untuk segitiga, bisa kita dapatkan :

$$B = \sqrt{{X_h}^2 + {Y_h}^2}, \qquad \text{ñ} = \tan^{-1}\left(\frac{X_H}{Y_H}\right)$$

$$\text{á} = \cos^{-1}\left(\frac{{L_1}^2 + B^2 - {L_2}^2}{2L_2B}\right), \quad \text{â} = \cos^{-1}\left(\frac{{L_1}^2 + {L_2}^2 - B^2}{2L_1L_2}\right)$$

Dari data area kerja robot, kita tahu L_1 = 125 mm, L_2 = 125 mm

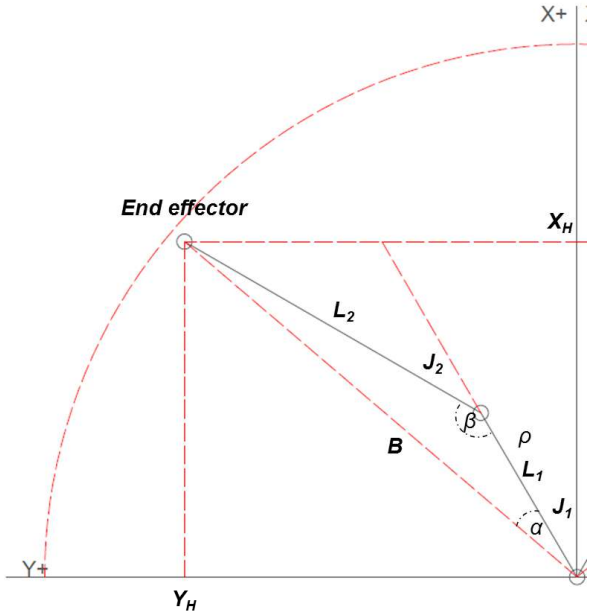
J_1 adalah selisih sudut ρ dan α, bertanda negatif karena terletak di kuadran X+ dan Y-

$$J_1 = -(\text{ñ} - \text{á}),\ J_1 = -\left(\tan^{-1}\left(\frac{X_H}{Y_H}\right) - \cos^{-1}\left(\frac{{L_1}^2 + {L_2}^2 - B^2}{2L_1L_2}\right)\right)$$

J_2 dilihat terhadap sumbu 0^0 L_1, adalah selisih sudut 180^0 dan $\boldsymbol{\beta}$, dan bertanda negatif karena terletak di kudaran sisi kanan L_1

$$J_2 = -(180 - \text{â}),\ J_2 = -\left(180 - \cos^{-1}\left(\frac{{L_1}^2 + {L_2}^2 - B^2}{2L_1L_2}\right)\right)$$

3.6.3 Penyelesaian persamaan *Inverse Kinematics* Kanan



- Bagian Kartesian :
  X_h = Posisi kerja dalam sumbu X
  Y_h = Posisi kerja dalam sumbu Y
- Bagian Sudut Servo :
  J_1 = Posisi sudut J1
  J_2 = Posisi sudut J2
- Definisi Tambahan :
  L_1 = Panjang *Inner-Link*
  L_2 = Panjang *Outer-Link*
  B = Diagonal antara X_h dan Y_h
  ρ = Sudut dalam antara X_h dan B
  α = Sudut dalam antara L_1 dan B
  β = Sudut dalam antara L_1 dan L_2

Gambar 3.12 Persamaan *Inverse Kinematics* Kanan

Dengan *theorema Phytagoras*, *trigonometri*, dan Cosinus untuk segitiga, bisa kita dapatkan :

$$B = \sqrt{{X_h}^2 + {Y_h}^2}, \qquad \text{ñ} = \tan^{-1}\left(\frac{X_H}{Y_H}\right)$$

$$\text{á} = \cos^{-1}\left(\frac{{L_1}^2 + B^2 - {L_2}^2}{2L_2B}\right), \quad \text{â} = \cos^{-1}\left(\frac{{L_1}^2 + {L_2}^2 - B^2}{2L_1L_2}\right)$$

Dari data area kerja robot, kita tahu L1 = 125 mm, L2 = 125 mm

J_1 adalah selisih sudut ρ dan α, bertanda positif karena terletak di kuadran X+ dan Y+

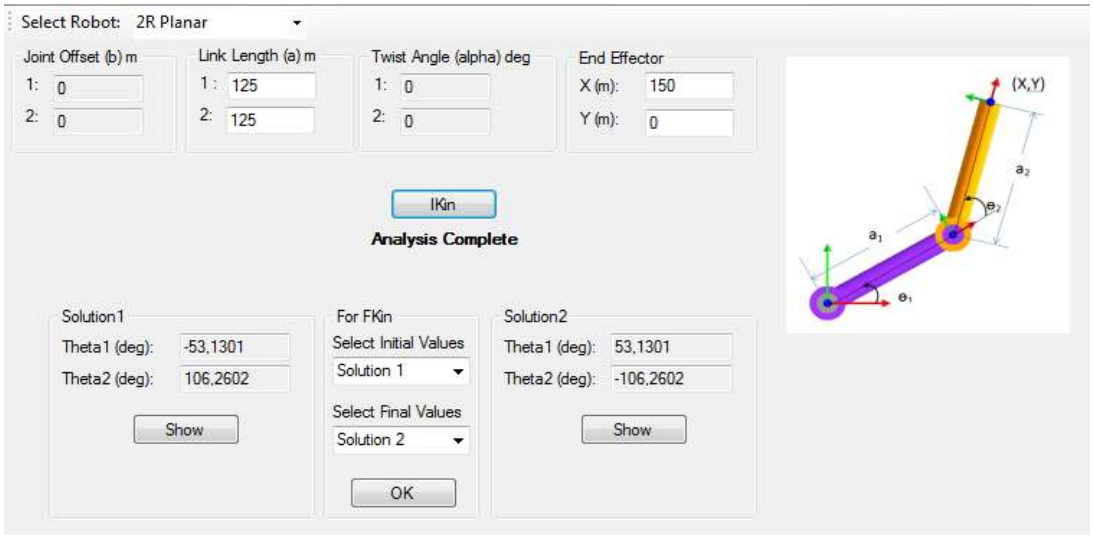
$$J_1 = \text{ñ} - \text{á}, \; J_1 = \tan^{-1}\left(\frac{X_H}{Y_H}\right) - \cos^{-1}\left(\frac{{L_1}^2 + {L_2}^2 - B^2}{2L_1L_2}\right)$$

J_2 dilihat terhadap sumbu 0^0 L_1, adalah selisih sudut 180^0 dan $\boldsymbol{\beta}$, dan bertanda positif karena terletak di kuadran sisi kiri L1

$$J_2 = 180 - \hat{a},\ J_2 = 180 - \cos^{-1}\left(\frac{{L_1}^2 + {L_2}^2 - B^2}{2L_1L_2}\right)$$

3.6.4 Simulasi *Inverse kinematic*

Berikut percobaan Simulasi inverse kinematic dengan memberikan nilai posisi pada *end effector* kedua link, dari simulasi didapatkan hasil 2 solusi yaitu solusi gerak kiri dan solusi gerak kanan



Gambar 3.13. Simulasi *inverse kinematic* dengan posisi *end effector* X150 mm dan Y0 mm

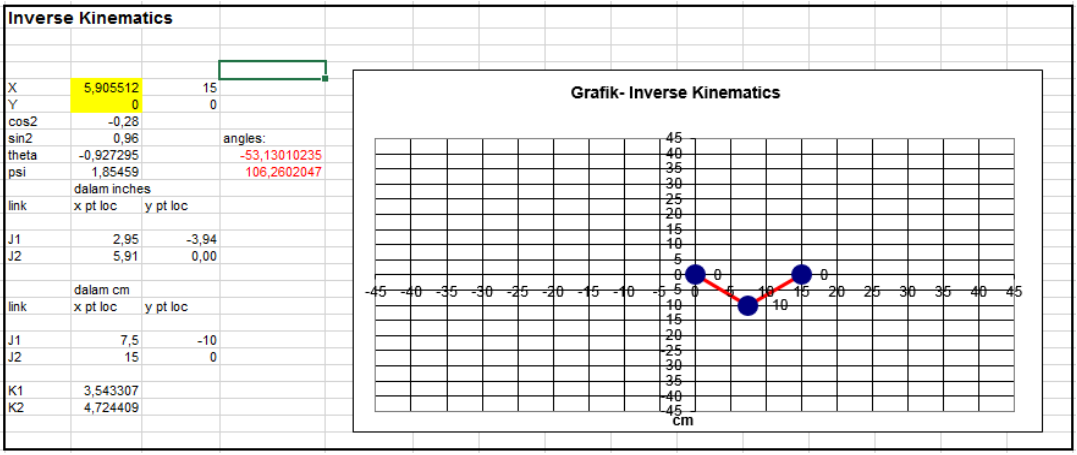
Matrik *inverse* memberikan informasi orientasi dan posisi dari *end-effector*. Kita lihat pada persamaan diatas posisi koordinat (Px, Py) *end-effector* dapat kita hitung dengan memberikan masukan posisi pada setiap *link*

Tabel 3.1 Percobaan posisi *End ffjector Robot manipulator*

No.	X	Y	Theta1	Theta2	Mode

1	150.00	0.00	53.130	-106.260	LEFTY
2	100.00	0.00	66.422	-132.844	LEFTY
3	150.00	0.00	-50.130	106.260	RIGHTY
4	100.00	0.00	-66.422	132.844	RIGHTY

Tabel 3.2. Perhitungan *inverse kinematic*



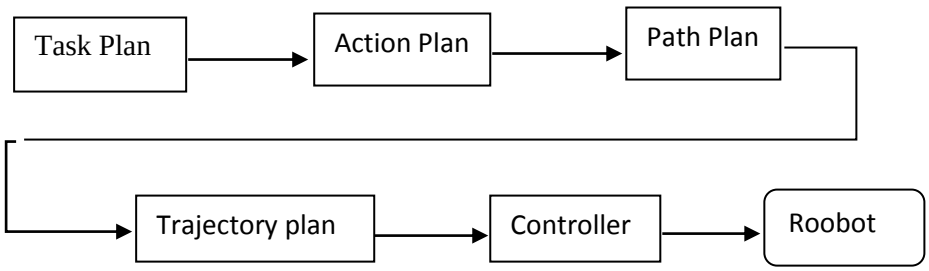
Pada tabel diatas untuk inverse kinematic untuk menentukan nilai sudut pada kedua sumbu lengan dan sudut teta dan *end-effector* terhadap sumbu x dan y seperti yang ditunjukkan pada gambar diatas.

3.7 Trajectory Planning

Seperti yang telah dijelaskan pada bab II, *joint trajectory planning* diperlukan agar *end-effector* dapat dengan tepat berpindah dari satu titik ke titik lain *(point to point motion).* Pada sistem manipulator ini banyak digunakan motor servo yang telah memiliki kendali posisi di dalamnya. Oleh karena itu pemilihan metode *pada joint space trajectory planning* bisa dipilih dengan bebas.

berikut ini *flowchart* yang digunakan untuk progarm Program *Path Planning* dengan interpolasi *linear* yang berupa bentuk garis lurus dan garis lingkaran

konsep trajectory planning



Gambar 3.14 Flowchart Program Path Planning

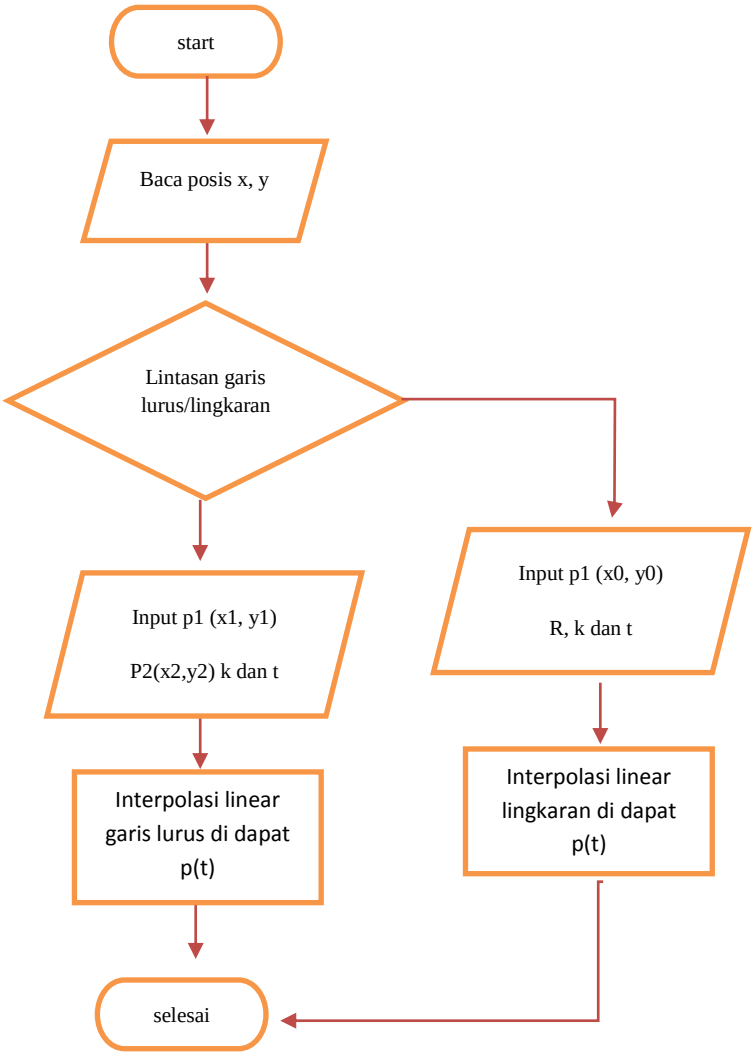
Path planning Geometric path

Path adalah ruang kurva dimana lengan robot (robot manipulator) yang diwakili oleh masing masing joint, bergerak dari posisi awal (*initial position*) menuju posisi akhir (*final Position*)

Trajectory planning

Interpolasi path yang di inginkan dengan fungsi *polynomial* dan membangkitkan nilai set point untuk pengendaliam robot dari posisi awal (*initial point)* menuju posisi yang diingionkan (*destination point*)

Ruang *variabel joint,* dimana *trajectory* di bentuk berdasarkan posisi. Ruang cartesian , dimana trajectory dibentuk berdasarkan posisi ujung lengan robot



Gambar 3.15 Flowchart Program *Path Planning* dengan *interpolasi linear*

3.7.1. Program *Path Planning* dengan Interpolasi *Linear*

Lintasan yang dilalui robot pada sistem ini dibatasi hanya berupa garis lurus atau lingkaran. Berikut adalah tahapan program untuk *path planning* dengan menggunakan interpolasi *linear* sesuai dengan persamaan pada bab 2.

Point to point motion lebih menitik-beratkan pada bagaimana perubahan posisi dan kecepatan di masing-masing sendi sehingga end effector dapat bergerak

dari titik awalnya menuju titik acuan dengan cepat dan tepat. *Joint space trajectory planning* merupakan metode kendali yang sangat tepat dan cepat untuk maksud ini. Berikut disimulasikan gerakan salah satu sendi manipulator baik menggunakan *linear trajectory planning* maupun dengan menggunakan *Cubic Polynomial Trajectory Planning*.

Simulasi gerak sendi menggunakan *linear trajectory planning*

Gerak sendi menggunakan *linear trajectory planning* disimulasikan dengan perpindahan sendi dari suatu titik ke titik lain tanpa perubahan kecepatan sama sekali. pada *linear trajectory planning* sendi akan berpindah secara *linear*.

BAB IV

ANALISA DATA DAN HASIL PERCOBAAN

4.1 Pengujian *Inverse Kinematics* Simulasi *Robot Manipulator*

Dalam permasalahan inverse kinematics hal yang akan dilakukan pengujian menitik beratkan pada bagian posisi, dimana yang diamati bagian *link 1* dan *link 2* pada *Robot Manipulator* 2-DOF untuk posisi awal dan ahkir *Robot Manipulator* . Posisi yang di dapat tersebut dapat dilihat pergerakannya posisinya dalam bentuk tabel perbandingan antara perhitungan dengan aktual. Dalam permasalahan inverse kinematics untuk mendapatkan solusinya menggunakan dengan pendekatan geometri. Dari metode tersebut akan didapatkan beberapa solusi pada masing-masing sudut joint robot. Dari metode geometri tersebut dibuatkan beberapa kondisi pergerakan *Robot Manipulator* .

4.2 Pengujian Pergerakan *Robot Manipulator*

Pengujian pergerakan *Robot Manipulator* bertujuan untuk mengetahui apakah sudut yang dikirim sesuai dengan pergerakan *Robot Manipulator* secara nyata. Untuk mempermudah pergerakan *Robot Manipulator*, maka akan dilakukan perbandingan gambar *Robot Manipulator* sebenarnya dan gambar *Robot Manipulator* tampak samping pada koordinat X dan Y.

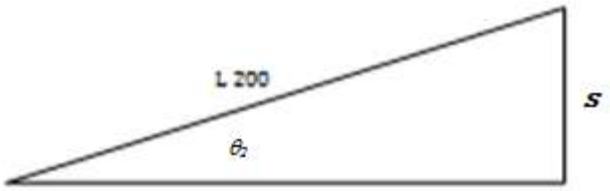
Tabel 4.1 Data Koordinat X dan Y dan Besar Pulsa Pada Tiap-Tiap Sendi

Pengujian	Koordinat Posisi		Sudut theta		Pulse	
	X	Y	*Theta1*	*Theta2*	Motor1	Motor2
1	250	0	0	0	0	0
2	200	0	-36.8699	73.7398	-6401.02	12802.05
3	150	0	-53.1301	106.2602	-9223.98	18447.95
4	100	0	-66.4218	132.8436	-11531.6	23063.13
5	50	0	-78.463	156.9261	-13622.1	27244.11

Pada pengujian pergerakan *Robot Manipulator* pada pengujian 1 posisi sudut theta dan pulse motor nilainya nol dikarenakan posisi tersebut merupakan posisi awal dari *Robot Manipulator*. Berikut dibawah ini merupakan gambar grafik dari pengujian 2 yaitu dengan posisi koordinat x200 dan y0 sehingga didapatkan nilai theta dan pulse yang dapat di lihat pada tabel diatas

Perhitungan akurasi dari gerakan lengan *Robot Manipulator* yang akan digunakan untuk menentukan penyimpangan yang terjadi dari setiap gerakan *Robot Manipulator*

Nilai akurasi dari motor Stepper = 0.00576 derajat



Sin(θ) S/L

S = 200 Sin200*SIN(0,00576/200*180/PI())*2

= 0,660047 mm

Untuk perhitungan gerak *Robot Manipulator* pada posisi koordinat X200 Y0 didapatkan nilai perhitungan

Dengan dasar perhitungan $sin\theta$ dan $cos\theta$

$$\cos\theta_2 = \frac{p_x^2 + p_y^2 - l_1^2 - l_2^2}{2l_1l_2}$$

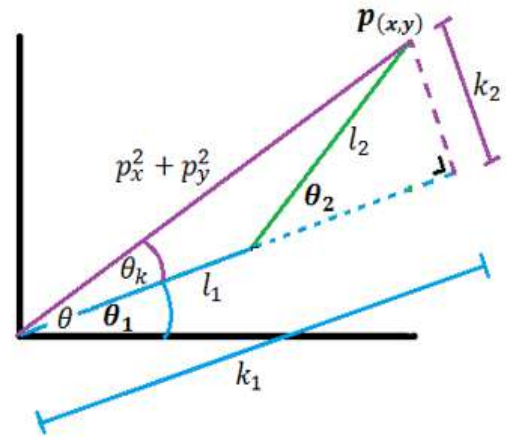
$Cos\theta_2$ = (200^2 + 200^2 – 125^2 – 125^2)/2 * 125 *125

= 0.28

$$sin\theta_2 = \pm\sqrt{1 - cos^2\theta_2}$$

$sin\theta_2$ = SQRT (1-0.28^2)

= 0.96



Dari nilai S$in\theta_2$ dan $cos\theta_2$ digunakan untukmencari nilai transformasi nilai K1 dan K2

K1 = $L1 + L2 * Cos\theta_2$

= 125 +125 * $Cos\theta_2$

= 250 * 0.28

= 160

K2 = L2 * $Sin\theta_2$

= 125 * 0.96

= 120

Untuk mencari *Theta1* dan theta 2 dilakukan dengan melakukan perehitungan dengan mengunakan nilai dari *K1* dan *K2*

$$\theta_1 = atan2 \quad (p_{yx}p_x) - atan2 \quad (k_{2x}k_1)$$

θ_1 = Atan2 (200*0) – Atan2 (120*160)

= -0.64

$$\theta_2 = tan2 \quad (sin\theta_{2\,x} cos\theta_2)$$

θ_2 = tan2(0.28*0.96)

= 1.28700

Dari nilai θ_1 dan θ_2 didapatkan nilai sudut yang akan digunakan untuk menggerakan motor1 dan motor2

Sudut *Theta1* (dalam derajat) = (-0.64 * 180)/PI

= -36.8699 derajat

Sudut *Theta2* (dalam derajat) = (1.28700 * 180)/PI

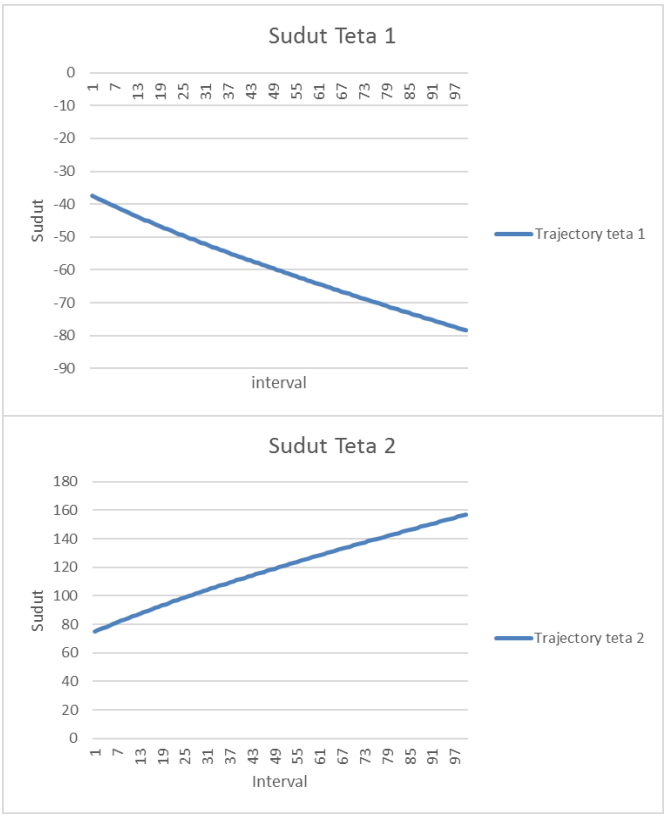
= 73.7398 derajat

Nilai Pulse motor1 = -36.8699 / 0.00576

= -6401.02 Pulse

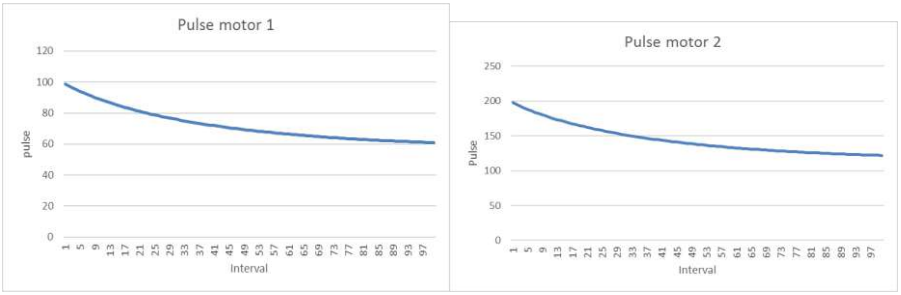
Nilai Pulse Motor2 = 73.7398 / 0.00576

= 12802.05 Pulse



Gambar 4.1 Grafik interval sudut teta 1 dan 2

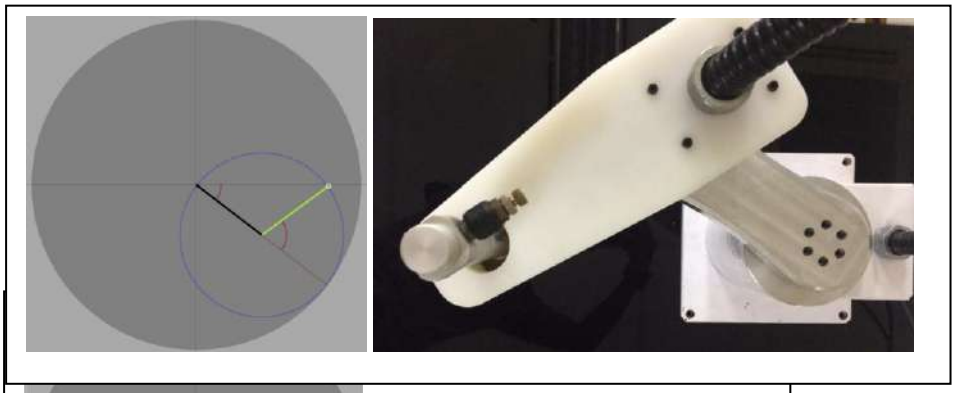
Berikut pulse motor1 dan motor2 yang digunakan untuk melakukan percobaan gerakpada *Robot Manipulator*.



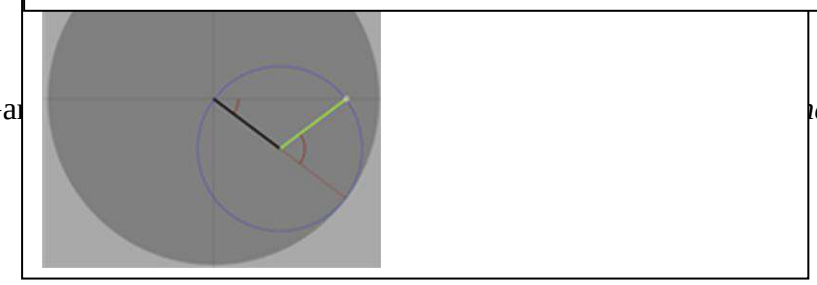
Gambar 4.2 Grafik interval pulse motor 1 dan 2

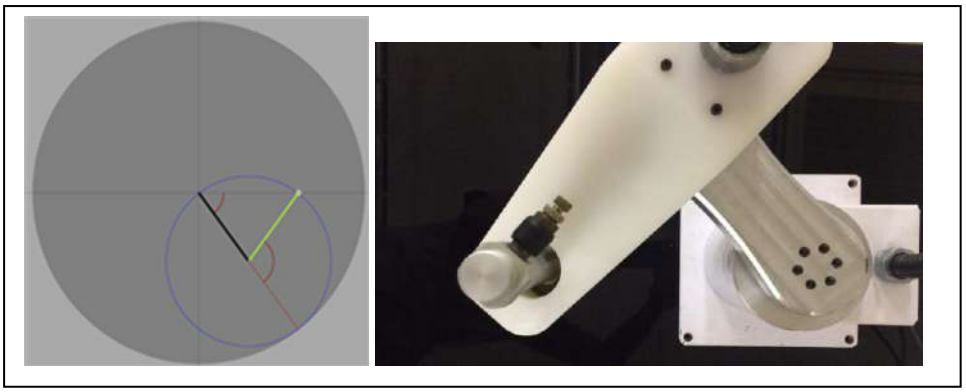
Pada percobaan dibawah ini akan di tampilkan gambar perbandingan gerak *Robot Manipulator* dari percobaan gerak pada koordinat sesuai pada tabel 4.1

Gambar 4.3. Simulasi dengan sudut gerak robot manipulator (*Theta1* 00 *Theta2* 00)

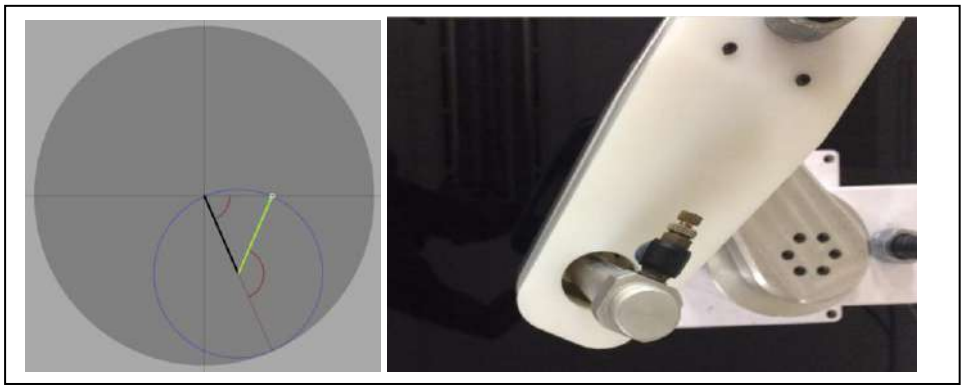


Ga *eta1*-35.620 *Theta2*

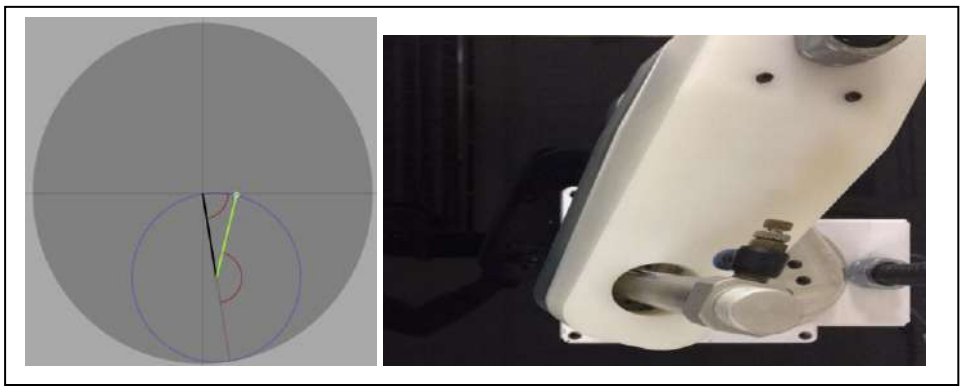




Gambar 4.5. Simulasi dengan sudut gerak robot manipulator (*Theta1* -51.465 *Theta2* 104.809.)



Gambar 4.6. Simulasi dengan sudut gerak robot manipulator (*Theta1*-64.390. *Theta2*. 131.575.)



Gambar 4.7. Simulasi dengan sudut gerak robot manipulator (*Theta1*-75.137 *Theta2* 155.727)

4.3. Pengujian Kesesuaian Perhitungan Dan Pergerakan *Robot Manipulator*

Pengujian kesesuaian perhitungan dan pergerakan *Robot Manipulator* berfungsi untuk mengetahui nilai error pada pergerakan *Robot Manipulator*. Pada penelitian ini dilakukan 5 (lima) kali percobaan untuk masing masing percobaan dilakukan perbandingan menggunakan alat kalibrasi *Faro Arm* untuk mengetahui perbedaan hasil perhitungan dan percobaan

Presentasi error masing-masing sumbu koordinat dan nilai sudut tiap lengan dapat

dihitung menggunakan persamaan sebagai berikut

$Err = (\Delta L/L) \times 100\%$

Keterangan:

Err = persentase *Error* koordinat (%)

ΔL = jarak perbedaan nilai koordinat antara *Robot Manipulator* sebenarnya dan perhitungan

L = panjang pergerakan robot lengan pada perhitungan.

Tabel 4.2 Data Pengujian Untuk Lengan 1

Pengujian	Setting		hasil lengan 1	
	Theta 1	posisi	Theta1	posisi
1	0	0	0	0
2	30	73.7398		
3	45	106.2602		
4	60	132.8436		
5	90	156.9261		

Tabel 4.2 Data Pengujian Untuk Lengan 2

Pengujian	Setting		hasil lengan 2	
	Theta 2	posisi	Theta2	posisi
1	0	0	0	0
2		73.7398		
3		106.2602		
4		132.8436		
5		156.9261		

Tabel 4.2 Data Pengujian Untuk Sudut tiap Lengan

Pengujian	Sudut Setting		Sudut hasil pergerakan	
	Theta 1	Theta 2	Link 1	Link 2
1	0	0	0	0
2	-36.8699	73.7398	-35.620	71.806
3	-53.1301	106.2602	-51.465	104.809
4	-66.4218	132.8436	-64.390	131.575
5	-78.463	156.9261	-75.137	155.727

Pengujian dengan memberikan nilai sudut pada *Robot Manipulator* dimana dari hasil percobaan akan di bandingkan dengan seberapa banyak selisih yang dihasilkan dari percobaan. Dari data tersebut di hitung presesntase Error pada *Robot Manipulator*.

Tabel 4.3 Data Nilai Presentase Error PadaTiap Sudut lengan

Pengujian	Selisih sudut Derajat		error (%)	
	ΔL *Theta1*	ΔL *Theta2*	ΔL *Theta1*	ΔL *Theta2*
1	000	000	000	000
2	1.250	1.934	3.390	2.693
3	1.665	1.451	3.140	1.384
4	2.032	1.469	3.058	0.964
5	3.727	1.199	4.238	0.770
Rata-rata Error			2.764	1.162

Keterangan:

$\Delta L\theta_1$ = Jarak perbedaan nilai antara robot lengan sebenarnya dan perhitungan menggunakan metode geometris.

$\Delta L\theta_2$ = Jarak perbedaan nilai antara robot lengan sebenarnya dan perhitungan menggunakan metode geometris.

Err θ_1 = Persentase *Error* pada θ_1

Err θ_2 = Persentase *Error* pada θ_2

Tabel 4.4 Data Pengujian Untuk Koordinat *End effector*

Pengujian	Sudut Setting		Koordinat hasil Perhitungan		Koordinat hasil Pergerakan	
	Theta 1	Theta 2	X	Y	X	Y
1	0	0	250	0	250	0
2	-36.8699	73.7398	200	0	200.96	1.29
3	-53.1301	106.2602	150	0	151.16	1.56
4	-66.4218	132.8436	100	0	103.13	1.73
5	-78.463	156.9261	50	0	55.15	1.94

Berdasarkan persamaan Diatas maka didapatkan nilai presentase *error* untuk *koordinat end*

*effecto*r pada Tabel 4.2 dan Nilai *presentase error* tiap sudut dapat dilihat pada Tabel 4.3.

Tabel 4. 5 Data Nilai Persentase *Error* Koordinat End effector

Pengujian	Selisih Koordinat (mm)		error (%)	
	ΔL X	ΔL Y	Error X	Error Y
1	0	0	0	0
2	0.96	1.29	1.92	1.29
3	1.16	1.56	2.32	1.56
4	3.13	1.73	6.26	1.73
5	5.15	1.94	10.3	1.94
Rata-rata Error			5.192	1.63

Keterangan:

ΔLX = Jarak perbedaan nilai koordinat X antara robot lengan sebenarnya dan perhitungan .

ΔLY = Jarak perbedaan nilai koordinat Y antara robot lengan sebenarnya dan perhitungan

Err X = Persentase *Error* pada koordinat X

Err Y = Persentase *Error* pada koordinat Y

4.4 Pengujian garis pada *Robot Manipulator*

Pengujian garis lurus dilakukan dengan memberikan nilai posisi pada setiap axis yaitu dengan menentukan titik koordinat x1,y1 sampai dengan x1y5 buat interval untuk di cari sudut teta1 dan teta 2 yang berikutnya di cari persamaan pulse yang akan dikirim robot simulator

Tabel 4. 6 Data Pengujian gerak pada 5 titik koordinat X dan Y

Pengujian	Hasil sudut perhitungan		Percobaan posisi		Koordinat hasil	
			Perhitungan		pergerakan	
	Theta 1	Theta 2	X	Y	X	Y
1	-36.8699	73.7398	180	0	180.16	1.46
2	-53.1301	106.2602	180	20	178.91	25.09
3	-66.4218	132.8436	160	20	159,83	25.74
4	-78.463	156.9261	160	0	158.71	2.25
5	-36.8699	73.7398	180	0	179.1	1.92

4.7 Data Nilai Persentase *Error* pada pengujian gerak pada 5 titik koordinat X dan Y

Pengujian	Selisih Koordinat (cm)		error (%)	
	ΔL X	ΔL Y	Error X	Error Y
1	0.16	1.46	0.08	1.46
2	1.09	5.09	0.6	5.09
3	0.17	5.74	0.1	5.74
4	1.29	2.25	0.8	2.25
5	0.90	1.92	0.5	1.92
Rata-rata Error			0.41	3.29

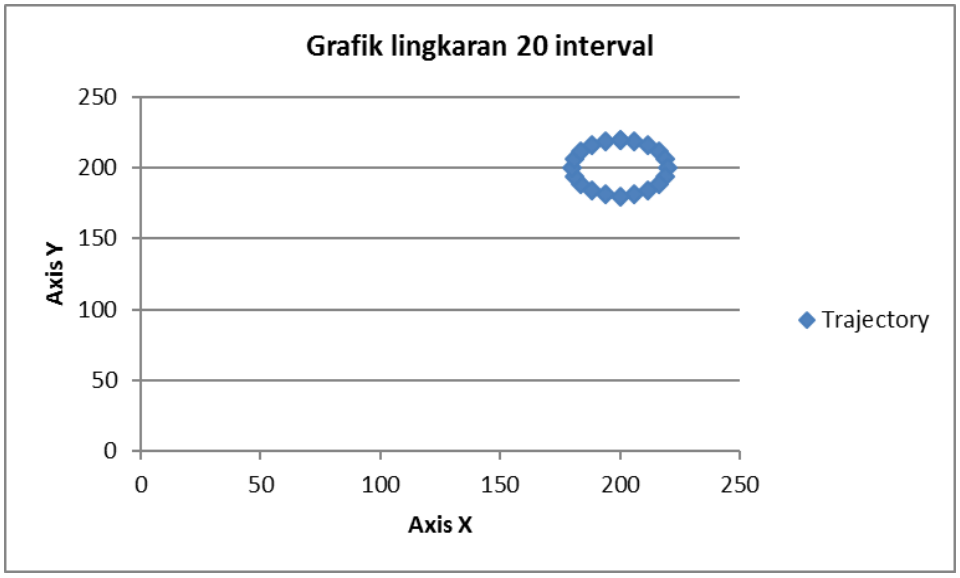
4.5 Pengujian garis Lingkaraan pada *Robot Manipulator*

Pada pengujian gerak garis lingkaran dengan menetapkan nilai koordinat pada X dan Y dimana nilai koordinat dari X dan Y tersebut di gunakan untuk titk

awal di buat garis lingkaran dengan radius yang di tentukan dan jumlah interval tertentu

Tabel 4.7 Data Pengujian gerak garis lingkaran dengan 20 interval pada X dan Y

Pengujian	Koordinat perhitungan (mm)		Koordinat gerak robot (mm)	
	X	Y	X	Y
1	220.00	0	220.75	0,5
2	219.021	6.18	219.68	5,96
3	216.18	11.756	215.68	11,73
4	211.756	16.18	210.82	16,48
5	206.18	19.021	209.79	20,54
6	200.00	20	198.84	23,23
7	193.82	19.021	192.96	21,57
8	188.244	16.18	187.83	17,8
9	183.82	11.756	184.44	12,76
10	180.979	6.18	181.87	10,65
11	180.00	0	180.21	1,16
12	180.979	-6.18	180.24	5,33
13	183.82	-11.756	180.24	10,65
14	188.244	-16.18	182.04	17,4
15	193.82	-19.021	185.24	20,97
16	200.00	-20	189.49	22,17
17	206.18	-19.021	195.35	20,17
18	211.756	-16.18	202.21	16,55
19	216.18	-11.756	208.99	11,34
20	219.021	-6.18	214.21	6,33
21	220	0	219.29	1,51



Gambar 4.8 Grafik *trajectory* garis lingkaran 20 *interval*

Pada pengujian gerak *Tracjetory* garis Lingkaran dengan 20 interval dengan tujuan untukmengetahui seberapa besarpenyimpangan yang terjadi dari pergerakan simulasi dengan pergerakan robot manipulator dimana dari hasil percobaan tersebut dapat diketahui seberapa besar tinggkat Error yang terjadi

4.8. Data Nilai Persentase *Error* pada pengujian gerak garis lingkaran dengan 20 interval pada X dan Y

interval	Selisih Koordinat (mm)		error (%)	
	ΔL X	ΔL Y	Error X	Error Y
1	0,750	0,500	0,340909	0
2	0,659	0,220	0,300884	3,55987055
3	-0,5	0,026	-0,23129	0,221163661
4	-0,936	0,300	-0,44202	1,854140915
5	3,61	1,519	1,750897	7,98591031
6	-1,16	3,230	-0,58	16,15
7	-0,86	2,549	-0,44371	13,40097787
8	-0,414	1,620	-0,21993	10,01236094
9	0,62	1,004	0,337286	8,540319837
10	0,891	0,910	0,492322	7,871972318
11	0,21	1,160	0,116667	0
12	-0,739	0,850	-0,40833	13,75404531
13	-3,58	1,106	-1,94756	9,407961892
14	-6,204	1,220	-3,29572	7,540173053
15	-8,58	1,949	-4,42679	10,24656958
16	-10,51	2,170	-5,255	10,85
17	-10,83	1,160	-5,25269	6,102051552
18	-9,546	0,370	-4,50802	2,286773795
19	-7,19	0,416	-3,32593	3,538618578
20	-4,811	0,150	-2,19659	2,427184466
21	-0,71	1,510	-0,32273	0
Rata-rata Error			1.723	6.4642

Hasil pengujian di dapat nilai error pada koordinat *end effector* dan juga nilai *error* pada tiap sudut lengan. Rata-rata nilai *error* pada titik koordinat X sebesar 5.192 %, Y sebesar 1.63%. Nilai error rata-rata titik sudut *Theta1* sebesar 2.764%, sudut *Theta2* sebesar 1.162% dan disebabkan oleh nilai sudut pada simulasi perhitungan dengan interface arduino terdapat pebedaan hal ini menyebabkan terjadinya error untuk beberapa titik koordinat.

Hasil pengujian pada gerak Garis pada 5 titik koordinat di dapatkan Rata-rata Error pada titik koordinat X sebesar 0.41%, Y sebesar 3.29.%. Dan pada pengujian gerak garis lingkaran dengan 20 interval didapatkan error rata-rata pada titik koordinat X sebesar 1.723 %, Y sebesar 6,464%.

Dari hasil penelitian diatas dapat di tariik kesimpulan dimana dari kesimpulan tesebut dapat di gunakan sebagai referensi untuk penelitian pengembangan berikutnya.

BAB V

KESIMPULAN DAN SARAN

5.1 Kesimpulan

- Akurasi robot manipulator ditentukan dengan perhitungan dengan panjang lengan dengan akurasi sudut *stepper motor* sebesar 0,00576 derajat yang menghasilkan linear sebesar 0,660047 mm
- Hasil pengujian Pada sudut tiap lengan manipulator di dapatkan Nilai Error sebesar *Delta Theta1* 2.764% *Delta Theta2* 1.162%
- Hasil pengujian untuk Koordinat *End Effector* didapatkan penyimpangan nilai *error* pada titik koordinat X sebesar 5.192 %, Y sebesar 1.63%
- Hasil Pengujian pada 5 titik (kotak persegi) koordinat X dan Y nilai penyimpangan error pada titik koordinat X sebesar 0.41%, Y sebesar 3.29.%.
- Hasil Pengujian gerak garis lingkaran dengan 20 interval pada koordinat X dan Y nilai penyimpangan pada titik koordinat X sebesar 1.723%, Y sebesar 6.464.%.

5.2 Saran

- Untuk meningkatkan ketepatan trajectory dalam percobaan dengan meingkatkan nilai akurasi sudut motor stepper lebih kecil dari 0,00576 derajat
- Untuk meningkatkan kemampuan perhitungan *aritmatic* yang presisi sebaiknya digunakan *interface* yang memeliki resolusi (kecepatan membaca dan menerima data) yang lebih tinggi
- Mengunakan metode rumus yang perhitungan yang lain untuk mencari solusi yang cocok dari nilai penyimpangan diatas (*Rumus Jacobian*)