TUGAS AKHIR – TE 141599

KINEMATIKA BALIK MANIPULATOR ROBOT DENSO DENGAN METODE *NEURAL NETWORK*

Tegar Wangi Arlean
NRP 2215105051

Dosen Pembimbing
Ir. Rusdhianto Effendie AK, MT.

DEPARTEMEN TEKNIK ELEKTRO
Fakultas Teknologi Elektro
Institut Teknologi Sepuluh Nopember
Surabaya 2017

FINAL PROJECT – TE 141599

INVERSE KINEMATICS DENSO ROBOT MANIPULATOR WITH NEURAL NETWORK

Tegar Wangi Arlean
NRP 2215105051

Supervisor
Ir. Rusdhianto Effendie AK, MT.

ELECTRICAL ENGINEERING DEPARTEMENT
Faculty of Electrical Technology
Institut Teknologi Sepuluh Nopember
Surabaya 2017

PERNYATAAN KEASLIAN
TUGAS AKHIR

Dengan ini saya menyatakan bahwa isi sebagian maupun keseluruhan Tugas Akhir saya dengan judul "**Kinematika Balik Manipulator Robot Denso dengan Metode *Neural Network***" adalah benar-benar hasil karya intelektual mandiri, diselesaikan tanpa menggunakan bahan-bahan yang tidak diijinkan dan bukan merupakan karya pihak lain yang saya akui sebagai karya sendiri.

Semua referensi yang dikutip maupun dirujuk telah ditulis secara lengkap pada daftar pustaka. Apabila ternyata pernyataan ini tidak benar, saya bersedia menerima sanksi sesuai peraturan yang berlaku.

Surabaya, Juli 2017

Tegar Wangi Arlean
NRP 2215 105 051

--- Halaman ini sengaja dikosongkan --

KINEMATIKA BALIK MANIPULATOR ROBOT DENSO DENGAN METODE *NEURAL NETWORK*

TUGAS AKHIR

Diajukan Guna Memenuhi Sebagian Persyaratan
Untuk Memperoleh Gelar Sarjana Teknik
Pada
Bidang Studi Teknik Sistem Pengaturan
Departemen Teknik Elektro
Fakultas Teknologi Elektro
Institut Teknologi Sepuluh Nopember

Menyetujui :

Dosen Pembimbing I

Ir. Rusdhianto Effendie AK, MT.
NIP. 19570424 198502 1 001

KEMENTERIAN RISET, TEKNOLOGI, DAN PENDIDIKAN TINGGI
INSTITUT TEKNOLOGI SEPULUH NOPEMBER
FAKULTAS TEKNOLOGI ELEKTRO
DEPARTEMEN
TEKNIK ELEKTRO

SURABAYA
JULI, 2017

--- Halaman ini sengaja dikosongkan --

KINEMATIKA BALIK MANIPULATOR ROBOT DENSO DENGAN METODE *NEURAL NETWORK*

Nama : Tegar Wangi Arlean
Pembimbing : Ir.Rusdhianto Effendie AK, MT.

ABSTRAK

Manipulator robot adalah mekanik elektronik dengan cara kerja menyerupai lengan manusia. Manipulator robot adalah peralatan yang sering digunakan di dalam bidang industri robot dan tersusun dari sendi (*joint*) , *link* , dan *end-effector*. Terdapat dua jenis sendi robot, yaitu sendi putar (*revolute joint*) dan sendi geser (*prismatic joint*). Kinematika maju (*forward kinematics*) dan kinematika balik (*inverse kinematics*) merupakan konsep perhitungan daerah kerja dari manipulator robot. Daerah kerja dari *forward kinematics* berupa ruang *joint*, sedangkan daerah kerja *inverse kinematics* berupa ruang *cartesian. Forward kinematics* dan *inverse kinematics* diterapkan pada manipulator robot Denso 6-DOF. Perhitungan *forward kinematics* menghasilkan posisi yang di inginkan oleh *end-effetor* dari masukan berupa nilai semua *joint,* sedangkan *inverse kinematics* menghasilkan nilai untuk setiap *joint* dari masukan berupa posisi *end-effector*. Untuk menyelesaikan permasalahan *inverse kinematics* yang mempunyai banyak solusi dibandingkan dengan *forward kinematics*, maka digunakan metode *neural network*. Pada *inverse kinematics neural network* dilakukan pengujian pola persegi pada ketelitian 0,00005 dan *learning rate* 0,00025 dengan jumlah 150 *neuron* menghasilkan *error* jarak x 1,49e-05 m , *error* jarak y 1,06e-05 m, dan *error* jarak z 2,93e-05 m.

Kata Kunci : manipulator robot, Denso, 6-DOF, *forward kinematics, inverse kinematics, neural network.*

--- Halaman ini sengaja dikosongkan --

INVERSE KINEMATICS DENSO ROBOT MANIPULATOR WITH NEURAL NETWORK METHOD

Name **: Tegar Wangi Arlean**
Supervisor **: Ir.Rusdhianto Effendie AK, MT.**

ABSTRACT

Robot manipulator is an electronic mechanic by means of work resembling a human arm. Robot manipulators are commonly used tools in the robot industry and are composed of joints, links, and end-effector. There are two types of robotic joints, namely the revolute joint and the prismatic joint. Forward kinematics and inverse kinematics are the calculation concepts of the working area of the robot manipulator. The working area of the forward kinematics is a joint space, while the inverse kinematics working area is cartesian space. Forward kinematics and inverse kinematics are applied to the Denso 6-DOF robot manipulator. The calculation of forward kinematics produces the desired position by the end-effetor of the inputs of the value of all joints, whereas the inverse kinematics yields the value for each joint of the input as the end-effector position. To solve the inverse kinematics problem which has many solutions compared with forward kinematics, then the neural network method is used. In the inverse kinematics neural network, a square pattern test of 0.00005 and learning rate 0.00025 with 150 neurons resulted in x 1.49e-05 m distance error, 1.06e-05 m distance error, and z-distance error of 2.93e-05 m.

Keyword : *robot manipulator, Denso, 6-DOF, forward kinematics, inverse kinematics, neural network.*

--- Halaman ini sengaja dikosongkan --

KATA PENGANTAR

Puji syukur bagi Allah SWT yang telah memberikan rahmat dan karunia-Nya sehingga penulis dapat menyelesaikan Tugas Akhir dengan judul **"Kinematika Balik Manipulator Robot Denso dengan Metode *Neural Network*"**. Tidak lupa shalawat dan salam semoga selalu tercurah kepada Nabi Muhammad SAW yang telah memberi tuntunan dan pencerahan kepada umat manusia. Tugas Akhir ini disusun guna memenuhi sebagian persyaratan menyelesaikan pendidikan S1 pada Bidang Studi Teknik Sistem Pengaturan, Jurusan Teknik Elektro, Fakultas Teknologi Elektro, Institut Teknologi Sepuluh Nopember Surabaya.

Pelaksanaan serta penyelesaian laporan Tugas Akhir tidak terlepas dari bimbingan, motivasi, masukan dan bantuan dari berbagai pihak. Untuk itu pada kesempatan ini penulis ingin menyampaikan ucapan terima kasih kepada:

1. Segenap keluarga, terutama Bapak dan Ibu tercinta serta adik penulis yang selalu memberikan dukungan, semangat dan doa sehingga penulis dapat menyelesaikan Tugas Akhir saat ini.
2. Bapak Ir.Rusdhianto Effendie AK, MT. sebagai dosen pembimbing penulis yang telah memberikan pengetahuan, arahan dan bantuan dalam menyelesaikan Tugas Akhir ini.
3. Rika Puspitasari Rangkuti sebagai seseorang yang selalu memberikan dukungan , semangat sekaligus sebagai partner dalam proses pengerjaan dan penyelesaian dari Tugas Akhir ini.
4. Seluruh teman alumni D3 Teknik Elektro 2012 , teman kos GW25C dan kos GH16A atas semua dukungan yang diberikan.

Penulis menyadari bahwa dalam penyusunan laporan Tugas Akhir ini masih terdapat kekurangan dan jauh dari kata sempurna. Untuk itu penulis mengharapkan saran dan kritik yang sifatnya konstruktif dalam penyempurnaan laporan ini. Akhir kata penulis berharap semoga laporan Tugas Akhir ini dapat bermanfaat bagi semua pihak, baik bagi diri penulis pribadi maupun pembaca.

Surabaya, Juli 2017

Penulis

--- Halaman ini sengaja dikosongkan --

DAFTAR ISI

--- Halaman ini sengaja dikosongkan --

DAFTAR GAMBAR

DAFTAR TABEL

--- Halaman ini sengaja dikosongkan --

BAB 1
PENDAHULUAN

Bab pendahuluan berisi tentang latar belakang, rumusan masalah, tujuan, dan metodologi pada tugas akhir tentang kinematika balik manipulator robot Denso dengan metode *Neural Network*.

1.1 Latar Belakang

Robotika adalah salah satu ilmu yang banyak digunakan di bidang industri untuk pengembangan teknologi yang baru. Robot melakukan beberapa tugas seperti memilih dan menempatkan suatu objek. Robot yang biasa dipakai dalam dunia industri adalah robot lengan. Lengan robot tersebut biasanya dikenal dengan manipulator robot.

Manipulator robot adalah sistem mekanik yang dapat melakukan beberapa tugas diantara lain memilih dan menempatkan objek sesuai dengan perintah yang diprogram. Manipulator robot sering digunakan dalam dunia industri untuk melakukan pekerjaan – pekerjaan yang berbahaya atau pekerjaan berulang – ulang dan memerlukan keakuratan. Dengan digunakannya manipulator robot pada bidang industri akan mengurangi jumlah biaya yang dikeluarkan perusahaan untuk ganti rugi akibat kecelakaan kerja karyawan, dan biaya sistem pengaman keselamatan kerja.

Pergerakan dari robot diatur sesuai perintah yang diberikan, yang biasanya adalah berupa letak objek di ruang kartesian. Untuk mengatur pergerakan robot sesuai perintah, maka diperlukan studi kinematika , dimana kinematika robot adalah studi analitis pergerakan lengan robot terhadap sistem kerangka yang diam/bergerak tanpa memperhatikan gaya yang mempengaruhi pergerakannya.

Karena sendi manipulator robot yang digunakan memiliki 6-DOF, maka pergerakan pada robot mempunyai banyak kemungkinan karena jumlah DOF lebih dari dua. Untuk mencari satu solusi kinematika balik dari banyak kemungkinan yang terjadi, maka diperlukan metode cerdas *Neural Network*.

1.2 Rumusan Masalah

Rumusan masalah yang dibahas pada tugas akhir ini adalah bagaimana menentukan *input* dan *output* dari *neural network* serta menemukan mekanisme formulasi untuk pembelajaran masing – masing

sendi sampai pembentukan struktur dari *neural network* untuk pendekatan solusi kinematika balik.

1.3 Batasan Masalah

Ada beberapa batasan yang terdapat dalam pembahasan tugas akhir ini, yaitu sebagai berikut:

1. Tugas Akhir ini membahas kinematika manipulator robot Denso 6-DOF yang memiliki konfigurasi RRR.
2. Hanya membahas kinematika tanpa pembahasan *velocity* dan dinamika.
3. Kinematika yang dibahas hanya melingkupi posisi *end-effector* tanpa pembahasan orientasinya.
4. Kinematika balik hanya mencari besar sendi tiga sendi awal untuk mencari posisi.
5. Tidak membahas perhitungan *singularity*.

1.4 Tujuan Penelitian

Tujuan yang hendak dicapai dalam tugas akhir ini adalah mendapatkan mekanisme perhitungan kinematika balik dengan menggunakan metode pendekatan *neural network* agar manipulator robot dapat menemukan posisi titik yang diberikan.

1.5 Metodologi

Metodologi yang digunakan dalam penyusunan tugas akhir ini antara lain:

1. Studi literatur

   Kegiatan dalam tugas akhir yang dikerjakan dengan mempelajari cara kerja dari manipulator robot. Setelah mengetahui cara kerja dan spesifikasi dari manipulator robot, maka dilanjutkan dalam mempelajari tentang bagaimana cara menentukan sendi dan koordinat *end-effector* manipulator robot dengan studi analisis kinematika. Setelah mempelajari kinematika robot dilanjutkan dengan mempelajari metode *Neural Network* yang akan digunakan sebagai penyelesaian kinematika balik untuk menemukan titik – titik koordinat yang ingin dicapai oleh robot.

2. Mencari Parameter
   Untuk mencari solusi pergerakan manipulator robot yang diinginkan, maka diperlukan untuk mencari parameter – parameter dari robot. Parameter – parameter dari robot akan digunakan untuk merancang kinematika maju (*forward kinematics*) yang berfungsi menetukan posisi awal atau posisi tujuan *end effector* yang diinginkan. Dari posisi awal yang dicari maka akan digunakan sebagai permasalahan kinematika balik (*inverse kinematics*).

3. Penyelesaian Kinematika Balik
   Hasil perhitungan kinematika maju (*forward kinematics*) akan digunakan sebagai penentuan titik awal dari *end effector* robot. Dari titik awal atau titik yang dituju akan dicari semua besar sudut sendi pada robot untuk mencapai titik tersebut menggunakan kinematika balik (*inverse kinematics*). Solusi pergerakan robot dari kinematika balik (*inverse kinematics*) dengan menngunakan metode *neural network* akan dicari dalam simulasi menggunakan *software* Matlab 2014,

4. Simulasi Pergerakan Manipulator Robot
   Setelah solusi kinematika balik (*inverse kinematics*) telah didapatkan, maka langkah selanjutnya adalah melakukan percobaan pergerakan pada simulasi yang dirancang dalam *software* Matlab 2014,

5. Penulisan Buku Tugas Akhir
   Buku tugas akhir ditulis secara intensif bila proses pengujian telah selesai. Pada saat proses pengujian sedang berjalan, dilakukan eksplorasi bahan-bahan untuk penulisan buku Tugas Akhir dari jurnal-jurnal ilmiah dan buku. Penulisan buku mempunyai 5 bab inti untuk ditulis yaitu, pendahuluan yang berisikan uraian singkat tentang tugas akhir, dasar teori yang berisikan materi – materi yang diperlukan dalam pengerjaan tugas akhir, perancangan metode untuk menyelesaikan kinematika balik (*inverse kinematics*), hasil analisa ,serta penutup yang berisiskan kesimpulan dan saran untuk tugas akhir yang telah dikerjakan.

1.6 Sistematika Penulisan

Pembahasan pada Tugas Akhir ini dibagi menjadi lima bab dengan sistematika penulisan sebagai berikut :

Bab I : **Pendahuluan**

Pada bab ini menjelaskan mengenai latar belakang, rumusan masalah, batasan masalah, tujuan penelitian, sistematika penulisan, serta relevansi.

Bab II : Teori Dasar

Bab ini menjelaskan mengenai tinjauan pustaka, konsep dasar dari robot, *forward kinematic, inverse kinematics* dan *neural network* untuk mendapatkan sudut pada masing-masing *joint* pada permasalahan *inverse kinematics*.

Bab III : Perancangan Sistem

Bagian ini berisi pembahasan tentang cara menentukan parameter DH manipulator robot Denso. Mencari Persamaan *forward kinematics* untuk mendapatkan posisi yang diinginkan. Pembentukan struktur *neural network* pada solusi *inverse kinematics* untuk mencari besar sudut *joint* yang diperlukan manipulator robot bergerak ke titik x,y,z target. Dari kedua Persamaan tersebut dibentuk desain simulasi *forwad kinematics* dan *inverse kinematics* dengan menggunakan *neural network* menggunakan *software* Matlab 2014 yang dilengkapi *toolbox Peter Corke* versi 10,

Bab IV : Pengujian dan Analisis Sistem

Bab ini memuat hasil simulasi beserta analisis data pada setiap pengujian *forward kinematics* dan *inverse kinematics* dengan menggunakan *neural network.*

Bab V : Penutup

Analisis yang dilakukan pada Bab IV akan diambil suatu kesimpulan. Saran diberikan sebagai bahan evaluasi penelitian kedepannya.

1.7 Relevansi

Hasil yang diperoleh dari tugas akhir ini diharapkan menjadi referensi dalam pengembangan sistem kontrol di bidang robotika pada manipulator robot. Selain itu, mengetahui fungsi dari metode cerdas *neural network* dapat digunakan sebagai solusi *inverse kinematics* untuk mencari besar sudut *joint* dari masukan titik x,y,z dalam ruang *cartesian*.

BAB 2
TEORI DASAR

Bab teori dasar berisi tentang pembahasan teori yang menunjang tugas akhir tentang manipulator robot serta bagiannya, baik itu jenis sendi sampai konfigurasinya. Dibahas juga mengenai kinematika maju untuk mencari persamaan posisi dan kinematika balik untuk mencari besar sudut yang dibutuhkan sendi, beserta metode *Neural Network* untuk menemukan solusi dari kinematika balik.

2.1 Tinjauan Pustaka [1]

Istilah robot pertama kali diperkenalkan kedalam kosa kata oleh dramawan Ceko Karel Capek pada tahun 1920 berupa kata "*robota*", yang berarti "bekerja" dalam kosa kata Ceko. Sejak saat itu istilah tersebut telah diaplikasikan untuk bermacam – macam perangkat mekanik, seperti teleoperator, kendaraan bawah air, dan lain sebagainya. Hampir semua yang beroperasi di bawah kendali dari komputer dapat disebut dengan robot. RIA (*Robot Institute of America*) mendefinisikan bahwa sebuah robot merupakan manipulator multifungsi bisa di program (*reprogrammable*) ulang yang didesain untuk memindahkan material, suku cadang, peralatan atau perangkat khusus dengan variabel gerakan yang diprogram untuk berbagai tugas.

Robot industri pada dasarnya berupa *mechanical arm* yang beroperasi di bawah kendali perangkat komputer. *Mechanical arm* atau manipulator robot memiliki dua bagian dasar yaitu lengan (*arm*) dan pergelangan (*wrist*). Manipulator robot tersusun dari lengan (*link*) yang dihubungkan dengan sendi (*joints*) menjadi sebuah rantai kinematika terbuka.

Ada dua jenis gerakan pada sumbu robot yang dapat menghasilkan pergerakan *link* yaitu sendi putar (*revolute joint*) dan sendi geser (*prismatic joint*). Serangkaian *prismatic joint* atau *revolute joint* disatukan untuk membentuk sebuah lengan manipulator yang mampu bergerak secara otomatis dalam jumlah derajat kebebasan (*-n- degree of freedom*).

Jumlah dari sendi dalam manipulator robot menentukan jumlah dari derajat kebebasan suatu robot. Manipulator robot dalam industri memiliki enam sendi, tiga sendi pertama digunakan untuk mengatur gerakan posisi robot. Tiga sendi berikutnya memberikan gerakan rotasi untuk

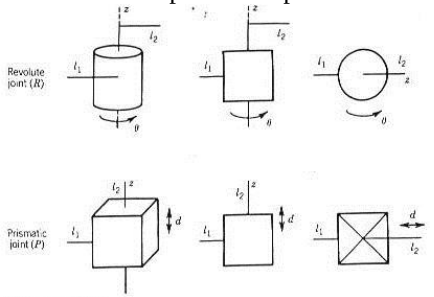
mengarahkan ujung manipulator robot (*end-effector*) untuk menghasilkan orientasi.

Untuk mengatur pergerakan manipulator robot sesuai posisi dan orientasi yang diinginkan, maka diperlukan analisa kinematika. Kinematika robot adalah studi analitis pergerakan lengan robot terhadap sistem kerangka yang diam/bergerak tanpa memperhatikan gaya yang mempengaruhi pergerakannya. Kinematika robot dapat dibedakan menjadi dua jenis yaitu kinematika maju (*forward kinematics*) dan kinematika balik (*inverse kinematics*).

Kinematika balik merupakan masalah utama dalam menentukan pergerakan robot, karena kinematika balik mengkonversi dari ruang *cartesian* ke ruang *joint*, dimana dapat terjadi banyak solusi yang terjadi. Kinematika balik dapat menjelaskan perhitungan sudut *joint* yang berhubungan dengan posisi dan orientasi *end-effector*. Dalam Tugas Akhir ini solusi kinematika balik akan diselesaikan menggunakan metode *Neural Network* untuk menentukan sudut yang diinginkan pada masing-masing *joint* dari manipulator robot. Desain untuk simulasi Kinematika maju dan Kinematika balik akan menggunakan batuan dari software Matlab 2014,

2.2 Manipulator Robot [1]

Struktur mekanis dari manipulator robot tersusun dari lengan (*link*) yang saling berhubungan melalui sendi (*joint*). Manipulator ditandai dengan sebuah lengan yang menjamin mobilitas dan pergelangan tangan yang diberikan kecepatan dan *end-effector* mengerjakan tugas yang diberikan kepada manipulator robot. *Joint* biasanya berbentuk *revolute* atau *prismatic*. Dalam hal ini dapat dilihat pada Gambar 2.1.



Gambar 2.1 Sendi *Revolute* dan *Prismatic*

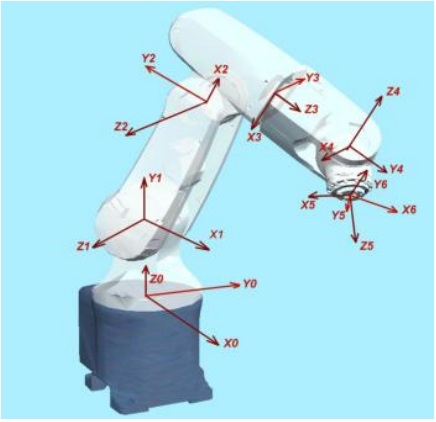
Hubungan antara dua *link* yang berurutan dapat dibentuk dari salah satu jenis *joint*, baik itu *prismatic* atau *revolute*. Setiap *joint prismatic* atau *joint revolute* memberikan struktur dengan satu derajat kebebasan (DOF).

Jumlah DOF tersebut merupakan jumlah sendi dalam menentukan DOF dari manipulator. Secara khusus, manipulator setidakanya memiliki 6-DOF, tiga untuk posisi dan tiga untuk orientasi. Jika kurang dari 6-DOF maka lengan tidak dapat mencapai setiap titik dalam lingkungan kerja dengan orientasi yang diinginkan.

Workspace (daerah kerja) manipulator menyatakan bahwa lingkungan manipulator *end-effector* bisa dijangkau. Daerah kerja dibatasi oleh geometri dari manipulator serta kendala mekanik pada *joint* manipulator robot.

2.2.1 Manipulator Robot Denso [2]

Salah satu tipe manipulator robot adalah manipulator robot Denso VP-6242G. Bentuk asli dari manipulator robot Denso VP-6242G dapat dilihat pada Gambar 2.2.



Gambar 2.2 Robot Denso VP-6242G

Prinsip gerakan manipulator robot Denso VP-6242G yang memiliki 6-DOF terbagi menjadi dua bagian, yaitu gerakan untuk menentukan posisi robot dan menentukan orientasi dari ujung manipulator robot. Posisi robot ditentukan berdasarkan gerakan 3 sendi pertama, sedangkan untuk menentukan orientasi didasarkan pada 3 sendi berikutnya.

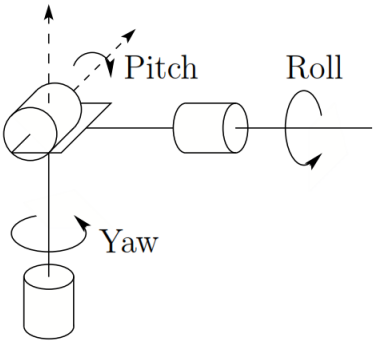
Keseluruhan sendi dari manipulator robot Denso VP-6242G merupakan sendi *revolute* yang berarti gerakan sendi yang paling mungkin adalah berotasi atau berputar.

2.3 Bagian Manipulator Robot [1]

Manipulator robot terdiri dari dua buah bagian inti yaitu pergelangan tangan (*wrist*) untuk memindahkan posisi dan *end-effector* untuk melakukan kerja.

2.3.1 *Wrist* [1]

Pergelangan tangan (*wrist*) dari manipulator robot mengacu pada *joint* dalam rangakaian kinematik antara lengan dan tangan. *Joint* pergelangan tangan hampir selalu *revolute*. Pergelagan tangan yang umum dipakai adalah *Spherical Wrist* yang berarti sumbu *joint* berpotongan di satu titik, kontruksinya dapat dilihat pada Gambar 2.3.

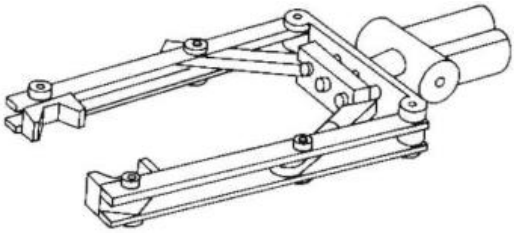


Gambar 2.3 Struktur dari *Spherical Wrist*

Spherical wrist sangat sederhana dalam analisis kinematik, karena memungkinkan untuk memisahkan analisa posisi dan orientasi. Lengan dan pergelangan tangan robot digunakan untuk menentukan posisi *end-effector* dan alat yang dibawanya.

2.3.2 *End of Effector* [1]

End-effector merupakan bagian manipulator robot yang melakukan kerja (*holding*, *drilling*, *weldding*). Jenis paling sederhana dari *end-effector* adalah *gripper*. Seperti yang ditunjukkan pada Gambar 2.4.



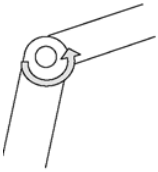
Gambar 2.4 *Gripper*

2.4 Jenis Sendi [3]

Sendi pada manipulator robot berfungsi untuk menghubungkan dua *link*, sehingga antara *link* tersebut dapat membentuk gerakan rotasi ataupun translasi sesuai bentuk sendi yang digunakan.

2.4.1 *Revolute Joint* [3]

Revolute Joint adalah sendi yang bergerak secara berputar / rotasi sesuai dengan batasan sudut gerakan yang dimiliki manipulator robot. Bentuk Umum dari *Revolute Joint* dapat dilihat pada Gambar 2.5.



Gambar 2.5 *Revolute Joint*

2.4.2 *Prismatic Joint* [3]

Prismatic Joint adalah sendi yang bergerak secara translasi sesuai dengan batasan gerakan yang dimiliki manipulator robot. Bentuk Umum dari *Prismatic Joint* dapat dilihat pada Gambar 2.6.

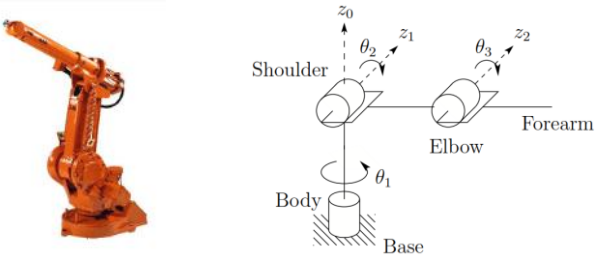
Gambar 2.6 *Prismatic Joint*

2.5 Konfigurasi Robot [4]

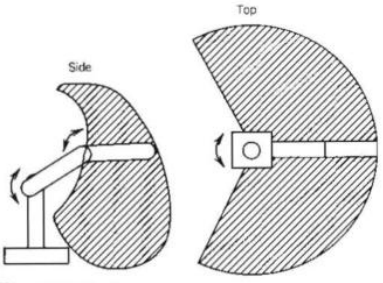
Berdasarkan penjelasan dari klasifikasi robot dilihat dari geometrinya dijelaskan bahwa terdapat lima konfigurasi robot,yaitu adalah sebagai berikut:

1. *Articulated Configuration* (RRR)

   Articulated manipulator sering juga disebut dengan *revolute*. Desain *revolute* yang paling sering digunakan adalah *elbow manipulator*, contohnya adalah robot ABB IRB1400, ditunjukkan pada Gambar 2.7. Dalam susunan *joint* seperti ini, *joint* sumbu z_2 paralel dengan sumbu z_1 , dan keduanya tegak lurus dengan sumbu z_0, Struktur dan istilah yang berkaitan dengan *elbow manipulator*. Sedangkan daerah kerjanya dapat dilihat pada Gambar 2.8. Konfigurasi ini memberikan manipulator robot kebebasan gerak yang sangat luas pada ruang yang diam.

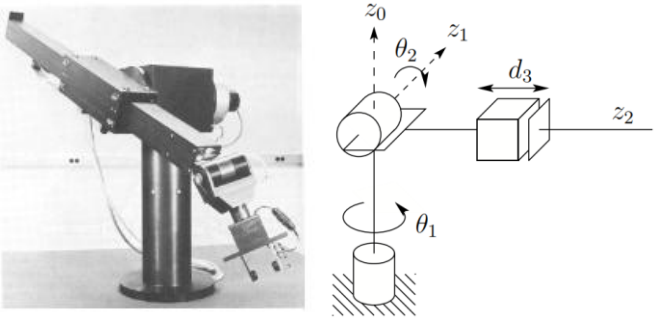


Gambar 2.7 ABB IRB1400 dan Struktur Konfigurasi RRR

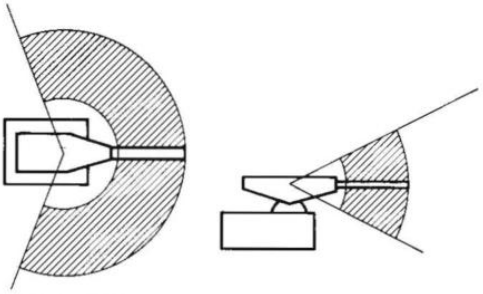


Gambar 2.8 Ruang Kerja dari *Articulated Configuration* (RRR)

2. *Spherical Configuration* (RRP)
   Dengan mengganti *joint* ke tiga dari *revolute configuration* dengan *prismatic joint* maka akan didapatkan *spherical configuration*, yang dapat dilihat pada Gambar 2.9. Istilah *Spherical configuration* dilahirkan dari fakta bahwa koordinat *spherical* menentukan posisi dari *end-effector*, dengan *frame* yang mempunyai titik *origin* hasil perpotongan sumbu z_1 dan z_2 pada 3 *joint* pertama. Manipulator robot *Stanford Arm* , robot yang memakai konfigurasi *spherical*. Sedangkan ruang kerjanya dapat dilihat pada Gambar 2.10.

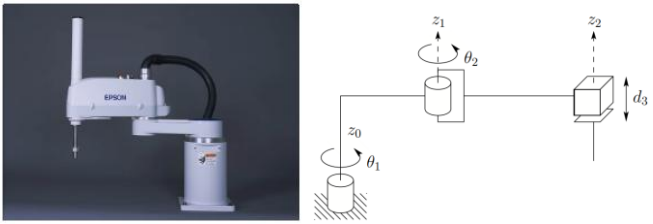


Gambar 2.9 *Stanford Manipulator* dan Struktur Konfigurasi RRP

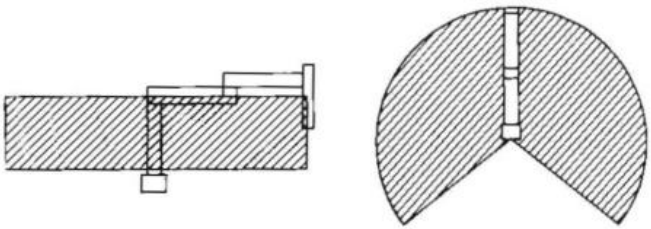


Gambar 2.10 Ruang Kerja dari *Spherical Configuration* (RRP)

3. *Scara Configuration* (RRP)
   SCARA (*Selective Compliant Articulatedd Robot for Assembly*) yang ditunjukkan pada Gambar 2.11 adalah konfigurasi yang populer saat ini, yang digunakan untuk operasi penataan (*assembly*). Walaupun SCARA mempunyai struktur RRP, *tetha*pi dalam penampilan dan *range* kerjanya sedikit berbeda dari *spherical configuration*. Bila *spherical configuration* memiliki z_0, z_1, z_2 yang saling tegak lurus, SCARA memiliki z_0, z_1, z_2 yang paralel. Robot yang menggunakan konfigurasi *Scara*. Sedangkan daerah kerjanya dapat dilihat pada Gambar 2.12.

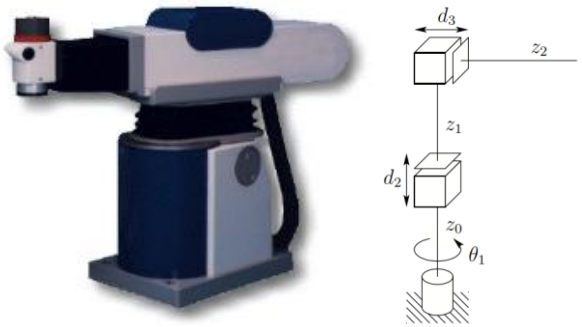


Gambar 2.11 *Epson* E2L653S dan Struktur dari SCARA (RRP)

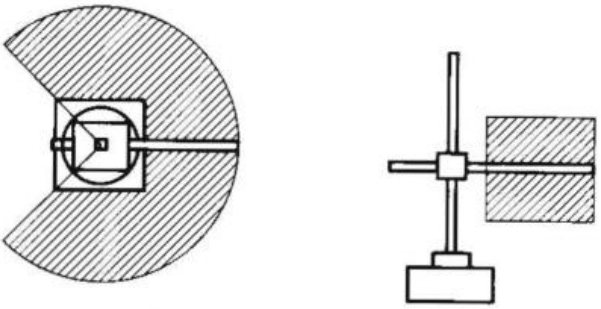


Gambar 2.12 Ruang Kerja dari *Scara Configuration* (RRP)

4. *Cylindrical Configuration* (RPP)
   Konfigurasi *cylindrical* ditunjukkan pada Gambar 2.13. *Joint* pertama adalah *revolute* dan menghasilkan rotasi terhadap bidang dasar (*base*), sedangkan *joint* kedua dan ketiga adalah *prismatic*. Dilihat dari namanya, variabel *joint*-nya merupakan *cylindrical coordinates* dari *end-effector* yang mengacu pada *base*. *Cyindrical* robot ditunjukkan daerah kerjanya Gambar 2.14.

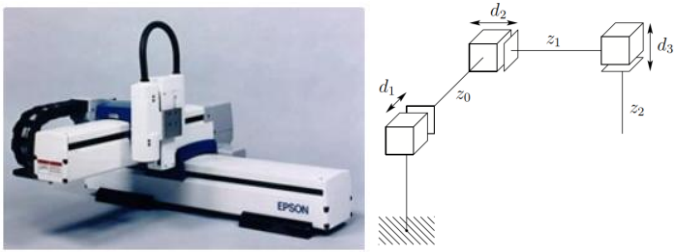


Gambar 2.13 *Seiko* RT3300 dan Struktur RPP

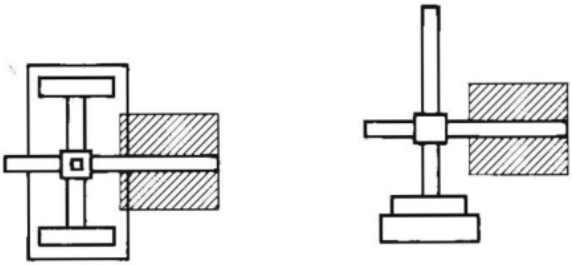


Gambar 2.14 Ruang Kerja dari C*ylindrical Configuration* (RPP)

5. *Cartesian Configuration* (PPP)
   Manipulator yang tiga *joint* pertama *prismatic* dikenal dengan *cartesian manipulator*, ditunjukkan pada Gambar 2.15. Untuk *cartesian manipulator*, *joint* variabelnya adalah koordinat *end-effector cartesian* yang saling berhubungan dengan *base*. Konfigurasi *cartesian* cocok digunakan sebagai penataan yang terdapat dalam meja atau untuk transfer material. Robot yang memakai konfigurasi *cartesian* dapat dilihat daerah kerjanya terdapat pada Gambar 2.16.



Gambar 2.15 *Epson Cartesian Robot* dan Struktur PPP



Gambar 2.16 Ruang Kerja dari *Cartesian Configuration* (PPP)

2.6 Transformasi Homogen[1]

Pergerakan rotasi dan translasi dari sebuah benda dapat direpresentasikan ke dalam transformasi homogen. Matriks gerakan translasi benda dapat dilihat pada Persamaan 2.1 dan rotasi benda dapat dilihat pada Persamaan 2.2.

$$Trans_{x,a} = \begin{bmatrix} 1 & 0 & 0 & a \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}; Trans_{y,b} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & b \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix};$$

$$Trans_{z,c} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & c \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (2.1)$$

Persamaan 2.1 untuk pergerakan translasi, dan

$$Rot_{x,\alpha} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & c_\alpha & -s_\alpha & 0 \\ 0 & s_\alpha & c_\alpha & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}; \qquad Rot_{y,\emptyset} = \begin{bmatrix} c_\emptyset & 0 & s_\emptyset & 0 \\ 0 & 1 & 0 & 0 \\ -s_\emptyset & 0 & c_\emptyset & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix};$$

$$Rot_{z,\theta} = \begin{bmatrix} c_\theta & -s_\theta & 0 & 0 \\ s_\theta & c_\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.2}$$

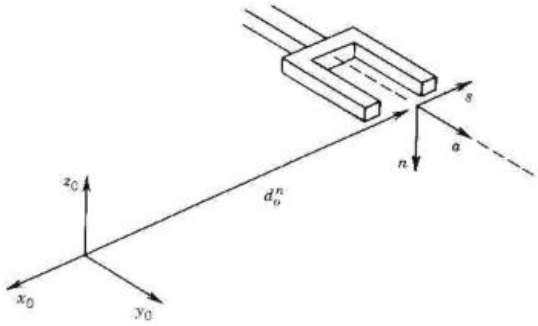
Persamaan 2.2 untuk rotasi terhadap sumbu x,y,z secara berurut. Bentuk umum dari transformasi homogen dapat ditulis dalam Persamaan 2.3.

$$H = \begin{bmatrix} n_x & s_x & a_x & d_x \\ n_y & s_x & a_x & d_x \\ n_z & s_x & a_x & d_x \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} n & s & a & d \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.3}$$

Pada Persamaan 2.3 **n** adalah vektoryang merepresentasikan vektor normal dari tangan, **s** adalah vektor geser dari tangan yang menunjuk dalam arah gerak jari sebagai *gripper* membuka dan menutup, **a** adalah vektor pendekatan dari tangan yang menunjuk ke arah normal terhadap telapak tangan dan **d** adalah vektor posisi dari tangan yang menunjuk dari titik asal dari sistem koordinat dasar menuju sistem koordinat tangan. Persamaan transformasi homogen dapat ditulis juga dalam bentuk Persamaan 2.4.

$$H_1^0 = \left[\begin{array}{c|c} R_{3x3} & P_{3x1} \\ \hline f_{1x3} & S_{1x1} \end{array}\right] = \left[\begin{array}{c|c} Rotation & Translation \\ \hline Perspective & Scale \end{array}\right] \tag{2.4}$$

Untuk lebih jelasnya Gambar 2.22 menunjukkan sistem koordinat tangan dan **n, s, a, d**.

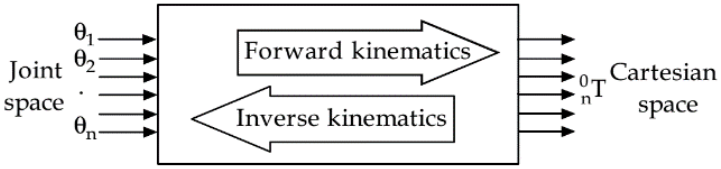


Gambar 2.17 Sistem Koordinat Tangan dan n,s,a,d

2.7 Kinematika Robot [5]

Kinematika adalah ilmu gerak yang memperlakukan subjek tanpa memperhatikan gaya yang menyebabkannya. Ilmu kinematika mempelajari tentang posisi, kecepatan, percepatan dan semua turunan yang lebih tinggi dari *variable* posisi.

Kinematika dibagi menjadi dua bagian yaitu kinematika maju (*forward kinematics*) dan kinematika balik (*inverse kinematics*). Hubungan antara *forward kinematics* dan *inverse kinematics* ditunjukkan pada Gambar 2.23.



Gambar 2.18 Alur *Forward kinematics* dan *Inverse kinematics*

Dimana *forward kinematics* merubah ruang besar sudut sendi ke ruang *cartesian* x,y,z , sedangkan *inverse kinematics* melakukan sebaliknya.

2.8 *Forward Kinematics* [1]

Sebuah permasalahan *forward kinematics* dapat dinyatakan dengan pernyataan sebagai berikut : Diberikan variabel *joint* dari manipulator robot , lalu tentukan posisi dan orientasi dari *end-effector*. Variabel *joint* adalah sudut diantara *link* apabila manipulator *revolute joint* atau sendi

putar, dan bisa juga merupakan panjang *link* apabila manipulator robot *prismatic joint* atau sendi geser. Serangkaian kinematik disebut terbuka ketika hanya ada satu urutan *link* yang menghubungkan kedua ujung rangkaian. Struktur mekanis manipulator ditandai oleh sejumlah derajat kebebasan (DOF).

2.8.1 *DH (Denavit-Hertenberg) Parameter* [1]

Dalam melakukan analisa kinematika diperlukan standarisasi penentuan koordinat dari sebuah *frame* pada setiap *link*. Standarisasi yang digunakan pada tugas akhir ini adalah pendekatan DH (*Denavit-Hertenberg*). Dalam pendekatan DH, setiap tansformasi homogen A_i direpresentasikan dengan perkalian dari empat transformasi dasar dapat dilihat pada Persamaan 2.5.

$$A_i = Rot_{z,\theta_i}\, Trans_{z,d_i}\, Trans_{x,a_i}\, Rot_{x,\alpha_i} \tag{2.5}$$

$$= \begin{bmatrix} c_{\theta_i} & -s_{\theta_i} & 0 & 0 \\ s_{\theta_i} & c_{\theta_i} & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & a_i \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & c_{\alpha_i} & -s_{\alpha_i} & 0 \\ 0 & s_{\alpha_i} & c_{\alpha_i} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} c_{\theta_i} & -s_{\theta_i} c_{\alpha_i} & s_{\theta_i} s_{\alpha_i} & a_i c_{\theta_i} \\ s_{\theta_i} & c_{\theta_i} c_{\alpha_i} & -s_{\theta_i} s_{\alpha_i} & a_i s_{\theta_i} \\ 0 & s_{\alpha_i} & c_{\alpha_i} & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Empat parameter $\theta_i, a_i, d_i, \alpha_i$ adalah parameter dari *link i* dan *joint* ke *i*. Parameter yang terdapat pada Persamaan 2.5 umumnya diberikan dengan nama: a_i disebut sebagai *length*, α_i disebut sebagai *twist*, d_i disebut sebagai *offset*, θ_i disebut sebagai *angle*.

Untuk memperoleh *forward kinematics* dari setiap manipulator terdapat prosedur berdasarkan standarisasi DH parameter. Berikut urutan algoritma yang harus dilakukan.

Langkah 1: Menempatkan dan melabeli *joint* sumbu $z_0, \ldots z_{n-1}$.

Langkah 2: Menetapakan *base frame*. Menentukan *origin* pada sumbu z_0. Sumbu x_0 dan z_0 dipilih secara acak untuk membentuk *right-hand frame*.

Langkah 3: Menempatkan *origin* O_i ke z_i dan z_{i-1} memotong z_i .

Langkah 4: Menetapkan x_i sepanjang *common* normal antara z_{i-1} dan z_i melalui o_i.

Langkah 5: Menetapkan y_i untuk melengkapi *right-hand frame*.

Langkah 6: Menetapkan *end-effector frame* pada $o_n x_n y_n z_n$.

Langkah 7: Membuat sebuah Tabel dari parameter *link* a_i, d_i, α_i, θ_i

a_i = jarak sepanjang x_i dari o_i ke perpotongan dari sumbu x_i dan z_{i-1}

d_i = jarak sepanjang z_{i-1} dari o_{i-1} ke perpotongan dari sumbu x_i dan z_{i-1}. d_i adalah variabel jika sendi i adalah sendi prismatik.

α_i = sudut antara z_{i-1} dan z_i diukur terhadap x_i.

θ_i = sudut antara x_{i-1} dan x_i diukur terhadap z_i. θ_i adalah variabel jika sendi i adalah *revolute*.

Langkah 8: Membentuk matriks transformasi homogen dengan melakukan substitusi parameter a_i, d_i, α_i, θ_i

Langkah 9: Membentuk matriks *forward kinematics* $T_0^n = A_1 \ldots A_n$ .Matriks ini memberikan posisi dan orientansi dari *tool frame* yang diekspresikan dalam koordinat dasar.

2.8.2 Menghitung *Forward Kinematics* [3]

Konsep dari kinematika maju adalah menghitung nilai tujuan dari sudut yang telah diberikan seperti diilustrasikan melalui Persamaan 2.6.

$$\theta = \theta_1, \theta_2, \theta_3, \ldots, \theta_n \quad (2.6)$$

Dari sudut – sudut yang telah diberikan tersebut akan didapatkan nilai akhir dari posisi robot. Nilai – nilai ini akan direpresentasikan sesuai matriks yang berisi seperti Persamaan 2.7, dimana notasi *n*, *s* dan *a* berisi nilai rotasi robot sedangkan notasi *d* berisi nilai translasi robot.

$$A = (n, s, a, d) \quad (2.7)$$

$$T_0^n = A_1 \ldots A_n \quad (2.8)$$

Sudut yang telah diberikan akan menjadi nilai akhir untuk posisi robot. Nilai akhir tersebut dimasukkan masing – masing ke Persamaan

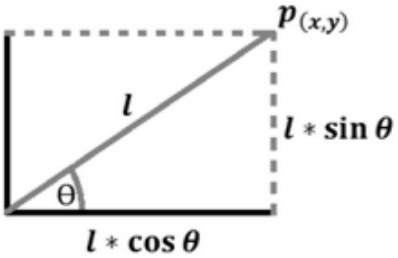
2.7 menggunakan perkalian dari matriks Transformasi Homogen untuk setiap link seperti yang tertera pada Persamaan 2.8.

2.9 *Inverse Kinematics* [5]

Kinematika terbalik (*inverse kinematic*) digunakan untuk menemukan variabel *joint* yang diperoleh dari posisi dan orientasi *end-effector*. Pada umumnya, perhitungan untuk *inverse kinematics* lebih sulit bila dibandingkan dengan *forward kinematics*. Karena permasalahan pada *inverse kinematics* adalah mencari sudut dari setiap sendi berdasar pada posisi akhir yang diinginkan sebuah robot, sehingga kinematika terbalik dikatakan solusi yang mempunyai hasil akhir yang unik. Dikatakan unik karena hasil dari sudut untuk tiap sendi ada banyak macam solusi tergantung dari banyaknya sendi yang digunakan. Karena itu pendekatan untuk mencari solusi *inverse kinematics* terdapat beberapa metode, diantaranya pendekatan numerik yang memanfaatkan perangkat komputer dan pendekatan geometri yang menganalisa dari ruang geometri manipulator robot.

2.9.1 Pendekatan Geometri [6]

Pendekatan geometri solusi dicari dengan menerapkan ilmu – ilmu geometri dan hukum – hukum trigonometri. Contoh seperti pada Gambar 2.24 yang menganalisa pada *link* pertama dari sebuah manipulator robot.



Gambar 2.19 Geometri *Link* Pertama Manipulator Robot

Dengan menggunakan Persamaan trigonometri : *Sinus* = depan/miring ; *cosinus* = samping/miring ; *tangen* = depan/miring, maka didapatkan:
Forward kinematics : (dimana besar θ diketahui dan posisi x,y dicari)

$$x = l * cos\theta \quad (2.9)$$

$$y = l * sin\theta \quad (2.10)$$

Inverse kinematics : (dimana posisi x,y diketahui dan besar θ dicari)
Dari Persamaan 2.9 dan 2.10 maka didapatkan, Persamaan 2.11 dan Persamaan 2.12.

$$cos\theta = x/l \quad (2.11)$$

$$sin\theta = y/l \quad (2.12)$$

Dari Persamaan 2.11 dan 2.12 akan didapatkan Persamaan 2.13 untuk mencari besar sudut *tetha*.

$$\theta = tan2^{-1}(sin\theta/cos\theta) \quad (2.13)$$

Persamaan 2.13 dapat ditulis dengan

$$\theta = Atan2(sin\theta, cos\theta) \quad (2.14)$$

Persamaan 2.13 datau 2.14 digunakan untuk mencari besar sudut sendi manipulator robot pada sendi pertama.

2.9.2 Pendekatan *Numeric* [5]

Pendekatan *numeric* adalah solusi dengan metode iteratif yang lebih umum *tetha*pi lebih lambat dan hanya dapat menemukan satu solusi untuk satu set nilai awal. Pendekatan *numeric* tidak mengutamakan diperoleh solusi yang tepat, *tetha*pi mengusahakan perumusan metode yang menghasilkan solusi pendekatan nilai yang dapat diterima. Pendekatan *numeric* memanfaatkan perangkat komputer untuk melakukan perhitungan secara berulang – ulang agar memperoleh solusi *inverse kinematics*. Komputer akan menghitung semua kemungkinan solusi secara berulang – ulang sampai diperoleh suatu solusi yang sesuai untuk sudut – sudut setiap *joint* yang dibutuhkan agar bisa mencapai posisi dan orientasi yang diinginkan.

2.10 *Neural Network* [7]

Jaringan saraf tiruan (*neural network*) merupakan suatu teknik pemrosesan informasi yang menggunakan model kuantitatif, jaringan saraf tiruan menghubungkan sejumlah masukan dan keluaran suatu sistem yang diorganisasikan dalam lapisan pemroses. Jaringan saraf tiruan atau JST terdiri atas elemen pemroses bernama *neuron*, yang dihubungkan dengan elemen pemroses lain oleh suatu aturan dan bobot. JST diGambarkan dengan bentuk grafik yang mempunyai arah menuju suatu

simpul dari elemen pemroses. Arah panah meneunjukkan arah normal suatu aliran sinyal. Pemrosesan sinyal di dalam jaringan dilakukan melalui proses komputasi. Secara garis besar, proses belajar JST dibagi menjadi 2:

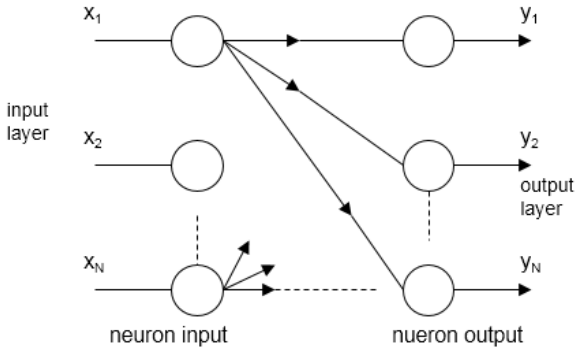
a. JST yang menggunakan paket pelatihan sebagai proses belajar dan dikenal sebagai proses belajar dengan pengawasan.
b. JST tanpa paket pelatihan pada proses belajar dan dikenal sebagai proses belajar tanpa pengawasan.

2.10.1 Arsitektur *Neural Network* [8]

Arsitekur jaringan saraf tiruan (*neural network)* adalah susunan *neuron* dalam *layer* (lapisan) dan pola koneksi untuk tiap neuron. Ada 2 jenis arsitektur *neural network* berdasarkan bentuk umum lapisannya, yaitu :

a. Jaringan *single layer*

Jaringan *single-layer* merupakan bagian dari jaringan *feedforward*, dimana sinyal datang dari input mengalir ke ouput.

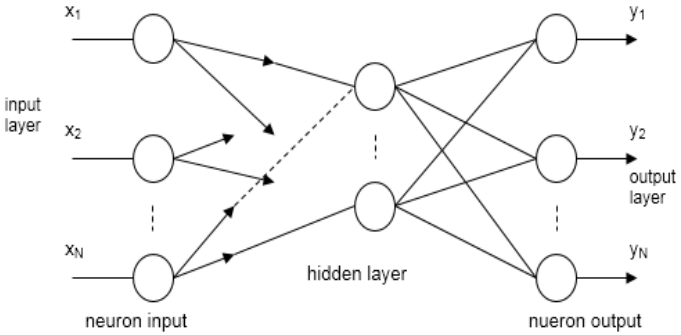


Gambar 2.20 *Single Layer Perceptron*

Jaringan *single layer* hanya mempunyai satu lapisan koneksi. Dalam sistem *single layer*, unit dapat dibedakan sebagai unit input dan unit output secara jelas. Biasanya dalam model single layer setiap unit input terhubung ke unit output *tetha*pi tidak terhubung ke unit input lainnya. Contoh *single layer* dapat dilihat pada Gambar 2.25.

b. Jaringan *multi layer*

Jaringan *multi layer* adalah jaringan yang mempunyai lebih dari satu lapisan *nodes* (titik hubung / koneksi) di antara input dan outputnya. Biasanya ada lapisan untuk menyatakan berat (*weight*) di antara dua level yang berdekatan. Jaringan *multi layer* dapat menyelesaikan permasalahan yang lebih kompleks. Contoh jaringan *multi layer* dapat dilihat pada Gambar 2.26.



Gambar 2.21 *Multiple Layer Perceptron*

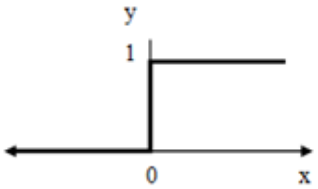
2.10.2 Jenis Fungsi Aktivasi *Neural Network* [8], [9]

Mengaktifkan *neural network* berarti mengaktifkan setiap *neuron* yang dipakai pada jaringan tersebut. Banyak fungsi yang dapat dipakai sebagai aktivasi, seperti fungsi unit step, impulse, sigmoid, dan lain sebagainya. Ada beberapa fungsi aktivasi yang sering digunakan dalam *neural network*, antara lain:

1. Fungsi *Hard Limit*

   Fungsi *hard limit* merupakan jaringan lapisan tunggal yang menggunakan *step function* untuk mengkonversikan *input* dari suatu variabel yang bernilai kontinyu ke suatu *output* biner (0 atau 1). Dapat dilihat pada Gambar 2.28 fungsi aktivasi *hard limit.*

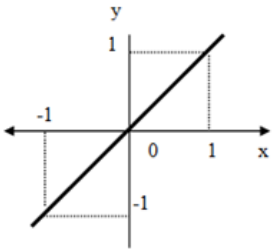
$$y = \begin{cases} 0, \text{jika } x \leq 0 \\ 1, \text{jika } x > 0 \end{cases}$$



Gambar 2.22 Fungsi Aktivasi *Hard Limit*

2. Fungsi *Linear* (Identitas)
   Fungsi ini memiliki nilai output yang sama dengan nilai inputnya, dapat dilihat pada Gambar 2.16 dan dapat dirumuskan sebagai berikut: dirumuskan y = x.

y = x

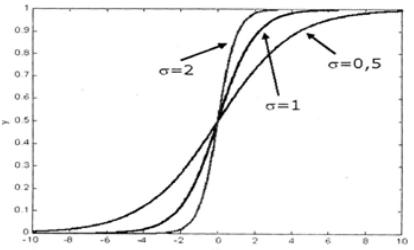


Gambar 2.23 Fungsi Aktivasi *Linear* (Identitas)

3. Fungsi *Sigmoid Biner*
   Dengan menggunakan metode *backpropagation*, mempunyai range 0 sampai 1, Biasanya digunakan untuk jaringan saraf yang membutuhkan nilai output yang terletak pada interval 0 sampai dengan 1, juga pada jaringan saraf yang nilai outputnya 0 atau 1, Fungsi ini dirumuskan sebagai berikut:

$$y = f(x) = \frac{1}{1+ e^{-\sigma x}}$$

dengan : $f'(x) = \sigma f(x)[1 - f(x)]$



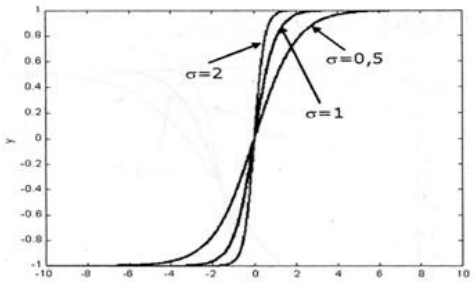
Gambar 2.24 Fungsi Aktivasi *Sigmoid Biner*

4. Fungsi *Sigmoid Bipolar*
   Fungsi ini hampir sama dengan fungsi *sigmoid biner*, *tetha*pi output fungsi ini memiliki range 1 sampai –1, Fungsi ini dirumuskan sebagai berikut:

$$y = f(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}}$$

atau $y = f(x) = \frac{1 - e^{-2x}}{1 + e^{-2x}}$

dengan : $f'(x) = [1 + f(x)][1 - f(x)]$



Gambar 2.25 Fungsi Aktivasi *Sigmoid Bipolar*

2.10.3 *Backpropagation Neural Network* [9]

Metode pembelajaran jaringan syaraf tiruan rambat balik (*Backpropagation Neural Network*) menggunakan ide perambatan balik

nilai *error* atau *generalized delta rule*. Delta rule merupakan metode *gradient descent* untuk meminimalkan jumlah kuadrat *error* keluaran. Pembelajaran dengan metode *backpropagation* meliputi tiga langkah, yaitu:

a. Perambatan maju (*feedforward*)
b. Perhitungan dan perambatan balik (*backpropagation*)
c. Penyesuaian nilai bobot berdasarkan *error*

Sedangkan Algoritma metode *backpropagation* adalah sebagai berikut:

Langkah 0 :
Pemberian inisialisasi pembobot (diberi nilai kecil secara acak)

Langkah 1 :
Untuk masing-masing pasangan data pelatihan (*training data*) lakukan langkah 2 hingga 7
Propagasi maju (Feedforward)

Langkah 2 :
Masing-masing unit *input* (x_i , i = 1,....n) menerima sinyal *input* x_i dan sinyal tersebut disebarkan ke unit-unit bagian berikutnya (unit-unit lapisan tersembunyi)

Langkah 3 :
Masing-masing unit dilapisan tersembunyi dikalikan dengan faktor pembobot dan dijumlahkan serta ditambah dengan biasnya:

$$z\_in_j = v_{oj} + \sum_{i=1}^{n} x_i v_{ij} \quad (2.15)$$

Kemudian menghitung sesuai dengan fungsi aktivasi yang digunakan:

$$z_j = f(z\_in_j) \quad (2.16)$$

Langkah 4 :
Masing-masing unit *output* (y_k, k=l,2,3,..m) dikalikan dengan faktor pembobot dan dijumlahkan:

$$y\_in_k = w_{ok} + \sum_{j=1}^{p} z_j w_{jk} \quad (2.17)$$

hitung kembali fungsi aktivasi

$$y_k = f(y\_in_k) \quad (2.18)$$

BackPropagasi dan errornya

Langkah 5:

Masing-masing unit *output* (y_k, k=l,...,m) menerima pola target sesuai dengan pola *input* saat pelatihan/*training* dan dihitung *error*nya:

$$\delta_k = (t_k - y_k) f'(y\_in_k) \quad (2.19)$$

Menghitung perbaikan faktor pembobot (kemudian untuk memperbaiki w_{jk}):

$$\Delta w_{jk} = \alpha \delta_k z_j \quad (2.20)$$

Menghitung perbaikan bias koreksi bias:

$$\Delta w_{0k} = \alpha \delta_k \quad (2.21)$$

dan menggunakan nilainya pada semua unit lapisan sebelumnya.

Langkah 6 :

Masing-masing pembobot yang menghubungkan unit-unit lapisan *output* dengan unit-unit pada lapisan tersembunyi (z_j,j=1,.,p) dikalikan *delta* dan dijumlahkan sebagai *input* ke unit-unit lapisan berikutnya.

$$\delta\_in_j = \sum_{k=1}^{m} \delta_k w_{jk} \quad (2.22)$$

Selanjutnya dikalikan dengan turunan dari fungsi aktivasinya untuk menghitung *error*nya.

$$\delta_j = \delta\_in_j f'(y\_in_j) \quad (2.23)$$

Kemudian menghitung perbaikan pembobot (digunakan untuk memperbaiki v_{ij}.

$$\Delta v_{ij} = \alpha \delta_j x_i \quad (2.24)$$

Kemudian menghitung perbaikan bias (untuk memperbaiki v_{0j})

$$\Delta v_{0j} = \alpha \delta_j \quad (2.25)$$

Langkah 7:

Masing-masing *output* unit (yk, k=1,..,m) diperbaiki bias dan pembobotnya (j=0,...,p)

$$w_{jk}(baru) = w_{jk}(lama) + \Delta w_{jk} \quad (2.26)$$

masing-masing unit tersembunyi (z_j, j: 1,....,p) diperbaiki bias dan pembobotnya (j=0,...,n).

$$v_{ij}(baru) = v_{ij}(lama) + \Delta v_{ij} \quad (2.27)$$

Langkah 8 :
Uji kondisi pemberhentian (akhir iterasi).

Langkah 9 :
Ulangi langkah 1 hingga 8 sampai kondisi akhir iterasi dipenuhi

Daftar Notasi
x_i = Unit *input* ke-i pada lapisan *input*
v_{0j} = nilai pembobot pada bias untuk unit z_j
v_{ij} = nilai pembobot dari unit x_i ke unit z_j
z_in_j = net *input* untuk z_j
z_j = Nilai aktivasi dari unit z_j
w_{0k} = nilai pembobot pada bias pada *output* unit y_k
w_{jk} = nilai pembobot dari z_j ke unit y_k
y_in_k = net *input* untuk y_k
y_k = *Output* unit ke-k pada lapisan *output*
t_k = *output* target dari *neural network*
δ_j = faktor pengaturan nilai pembobot sambungan pada lapisan *hidden*
δ_k = faktor pengaturan nilai pembobot sambungan pada lapisan *output*
α = konstanta laju pembelajaran (*leaming rate*) 0<<1
Δw_{jk} = selisih antara $w_{jk}(t)$ dengan $w_{jk}(t+1)$
Δv_{ij} = selisih antara $v_{ij}(t)$ dengan $v_{ij}(t+1)$

--- Halaman ini sengaja dikosongkan --

BAB 3
PERANCANGAN SISTEM

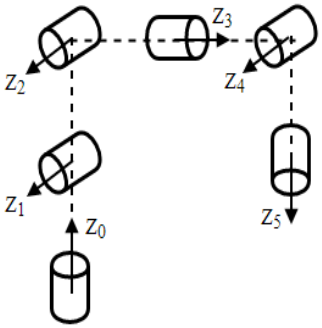
Pada Bab ini membahas tentang langkah dalam menentukan parameter DH (*Denavit Hartenberg*) yang nantinya akan digunakan untuk mencari Persamaan *forward kinematics.* Perancangan dan perhitungan *forward kinematics* merupakan patokan posisi yang menjadi *input* (masukan) pada *inverse kinematics neural network*.

Solusi *inverse kinematics neural network* dibentuk dengan cara menentukan struktur awal *neural network.* Struktur yang terbentuk akan digunakan untuk mencari solusi *inverse kinematic* dari manipulator robot. Perancangan solusi *inverse kinematics neural network* manipulator robot Denso akan disimulasikan dengan *software* Matlab dengan tambahan *toolbox Peter Corke* versi 10.

3.1 Menentukan Parameter DH (*Denavit Hartenberg*)

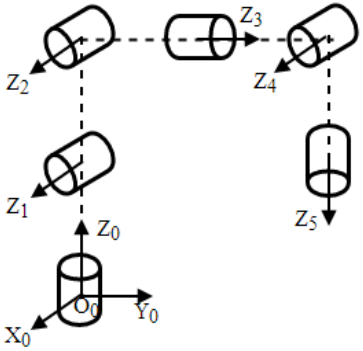
Dalam menentukan *inverse kinematics* , terlebih dahulu harus diketahui koordinat tujuan posisi yang ingin dicapai manipulator robot. Untuk membuat korrdinat dalam ruang *cartesian* dengan masukan besar sudut *joint* diperlukan perhitungan *forward kinematic*. Persamaan *forward kinematics* akan dicari menggunakan standarisasi aturan parameter DH (*Denavit Hartenberg*). Berikut adalah langkah – langkah untuk mencari parameter DH.

Langkah 1: Menentukan dan memberikan label pada sumbu Z_0 sampai Z_{n-1} seperti pada Gambar 3.1. dimana n adalah jumlah *joint* robot yaitu 6 sehingga sumbu yang ditentukan dari Z_0 sampai Z_5,



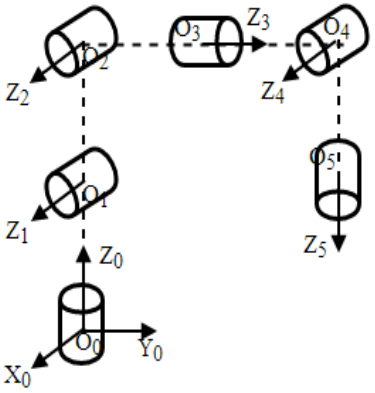
Gambar 3.1 Pemberian Label Sumbu z

Langkah 2:Menetapkan *base frame* dan menentukan titik *origin* (O_0) dimana saja sepanjang sumbu Z_0, selanjutnya melengkapi sumbu X_0 dan Y_0 sesuai kaidah tangan kanan seperi pada Gambar 3.2. *Base frame* ini nantinya digunakan sebagai acuan posisi dan orientasi dari *end-effector*.



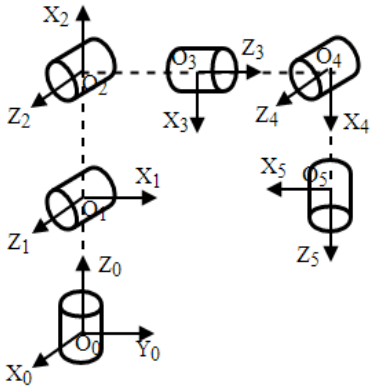
Gambar 3.2 Pemberian Label x dan y pada Sumbu Awal

Langkah 3: Menentukan titik O_i (untuk i=1 sampai i = n-1), dengan aturan jika sumbu Z_i memotong sumbu Z_{i-1}, maka titik O_i ditempatkan pada perpotongannya dan jika sumbu Z_i sejajar dengan sumbu Z_{i-1}, maka titik O_i ditempatkan pada *joint* i+1, Ilustrasi langkah ini ditunjukkan pada Gambar 3.3.



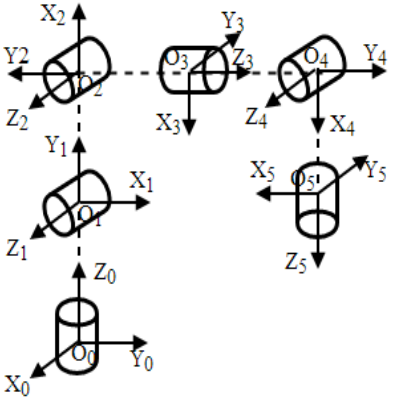
Gambar 3.3 Pemberian Label *Origin*

Langkah 4: Menentukan dan memberi label sumbu X_i (untuk i=1 sampai i = n-1), sepanjang *common normal* antara Z_{i-1} dan Z_i melalui O_i, atau pada arah normal ke bidang Z_{i-1} - Z_i jika Z_{i-1} dan Z_i berpotongan. Ilustrasi langkah ini ditunjukkan pada Gambar 3.4.



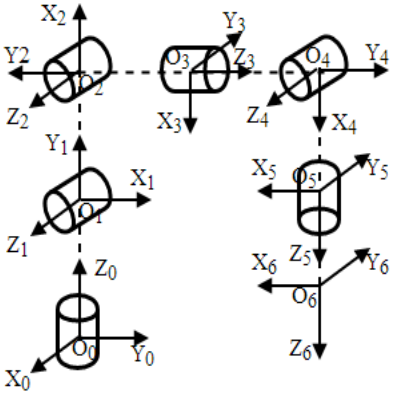
Gambar 3.4 Melengkapi Sumbu x

Langkah 5: Melengkapi sumbu Y_i (untuk i=1 sampai i=n-1), dengan aturan kaidah tangan kanan seperti pada Gambar 3.5.



Gambar 3.5 Melengkapi Sumbu y

Langkah 6: Menentukan *frame end-effector* $O_nX_nY_nZ_n$, dengan mengasumsikan bahwa *joint* ke n adalah *revolute* yaitu *joint* 6 dan arah sumbu Z_n yaitu Z_6 mengikuti sumbu Z terakhir (Z_5), selanjutnya melengkapi titik *origin* O_6 dan X_6 ,Y_6 dengan menggunakan kaidah tangan kanan seperti ditunjukkan pada Gambar 3.6.



Gambar 3.6 Menentukan *Frame End-Effector*

Langkah 7: Menentukan dan membuat Tabel parameter DH seperti ditunjukkan pada Tabel 3.1. Parameter yang dicari, yaitu :

Tabel 3.1 Parameter DH Manipulator Robot Denso [8]

Link	a_i	α_i	d_i	θ_i	*Range*
1	0	90	125	θ_1^*	-160 s/d 160
2	210	0	0	θ_2^*	-120 s/d 120
3	0	-90	0	θ_3^*	20 s/d 160
4	0	90	122	θ_4^*	-160 s/d 160
5	0	-90	0	θ_5^*	-120 s/d 120
6	0	0	70	θ_6^*	-360 s/d 360

Untuk memudahkan perhitungan mencari Persamaan *forward kinematics* , maka parameter DH manipulator robot Denso dibuat lagi

dengan mengumpamakannnya sebagai suatu variabel. Tabel parameter DH yang diubah ke dalam parameter dapat dilihat dalam Tabel 3.2.

Tabel 3.2 Variabel Parameter DH

Link	a_i	α_i	d_i	θ_i	*Range*
1	0	90	d_1	θ_1^*	-160 s/d 160
2	a_2	0	0	θ_2^*	-120 s/d 120
3	0	-90	0	θ_3^*	20 s/d 160
4	0	90	d_4	θ_4^*	-160 s/d 160
5	0	90	0	θ_5^*	-120 s/d 120
6	0	0	d_6	θ_6^*	-360 s/d 360

Variabel untuk besar sudut θ_1 sampai θ_6 memiliki tanda * pada atasnya yang menyatakan bilangan tersebut merupakan variabel yang bisa berubah – ubah dengan batas yang terdapat pada kolom *range* , dan sekaligus menyatakan pada sendi tersebut menggunakan sendi *revolute*.

3.2 Membentuk Representasi DH (*Denavit Hartenberg*)

Untuk merepresentasikan nilai dan rotasi dari sebuah manipulator robot dibutuhkan suatu transformasi homogen. Transformasi Homogen adalah matriks untuk merepresentasikan nilai translasi dan rotasi dari sebuah *link* terhadap sumbu *x, y, z*. Rotasi transformasi homogen pada manipulator robot merupakan arah putaran terhadap sumbu x,y,z pada robot. Translasi merupakan titik x,y,z tujuan yang dituju oleh *end-effector* manipulator robot. Persamaan transformasi homogen dapat dilihat pada Persamaan 2.5 pada bab 2 dan Persamaan tersebut direpresentasikan ke masing-masing *link* pada manipulator robot Denso dengan hasil sebagai berikut :

$$A_1 = \begin{bmatrix} c_1 & 0 & s_1 & 0 \\ s_1 & 0 & -c_1 & 0 \\ 0 & 1 & 0 & d_1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (3.1) \qquad A_4 = \begin{bmatrix} c_4 & 0 & s_4 & 0 \\ s_4 & 0 & -c_4 & 0 \\ 0 & 1 & 0 & d_4 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (3.4)$$

$$A_2 = \begin{bmatrix} c_2 & -s_2 & 0 & a_2c_2 \\ s_2 & c_2 & 0 & a_2s_2 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (3.2)$$

$$A_3 = \begin{bmatrix} c_3 & 0 & -s_3 & 0 \\ s_3 & 0 & c_3 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (3.3)$$

$$A_5 = \begin{bmatrix} c_5 & 0 & -s_5 & 0 \\ s_5 & 0 & c_5 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (3.5)$$

$$A_6 = \begin{bmatrix} c_6 & -s_6 & 0 & 0 \\ s_6 & c_6 & 0 & 0 \\ 0 & 0 & 1 & d_6 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (3.6)$$

Pada Persamaan 3.1-3.6 merupakan transformasi homogen manipulator robot Denso yang dihitung dari *link* satu sampai *link* enam. Hasil perhitungan tiap *link* akan dimasukkan untuk mencari Persamaan *forward kinematics*.

3.3 Mencari Persamaan *Forward Kinematics*

Forward kinematics merupakan permasalah dalam menentukan posisi akhir dari *end-effector*. Pada tahap ini akan menghitung nilai posisi dari *end-effector* dari manipulator robot Denso. Nilai transformasi homogen tiap *link* telah diperoleh pada Persamaan 3.1-3.6 sehingga dihitung *forward kinematics*. Perhitungan *forward kinematics* menggunakan Persamaan 3.8.

$$T_0^n = A_1\, A_2\, A_3 \dots \dots \dots A_n \quad (3.7)$$

Pada Persamaan ini akan dibuat Persamaan *forward kinematics*, dimana hasil dari perhitungan tersebut akan menghasilkan nilai rotasi dan posisi dari manipulator robot. Posisi yang didapat tersebut merupakan posisi yang akan dituju oleh *end-effector* dari manipulator robot Denso. Perhitungan *forward kinematics* ini dapat dilihat pada Persamaan 3.8.

$$T_0^6 = (A_1)(A_2)(A_3)(A_4)(A_5)(A_6) \quad (3.8)$$

$$T_0^6 = \begin{bmatrix} c_1 & 0 & s_1 & 0 \\ s_1 & 0 & -c_1 & 0 \\ 0 & 1 & 0 & d_1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} c_2 & -s_2 & 0 & a_2c_2 \\ s_2 & c_2 & 0 & a_2s_2 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} c_3 & 0 & -s_3 & 0 \\ s_3 & 0 & c_3 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\begin{bmatrix} c_4 & 0 & s_4 & 0 \\ s_4 & 0 & -c_4 & 0 \\ 0 & 1 & 0 & d_4 \\ 0 & 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} c_5 & 0 & -s_5 & 0 \\ s_5 & 0 & c_5 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} c_6 & -s_6 & 0 & 0 \\ s_6 & c_6 & 0 & 0 \\ 0 & 0 & 1 & d_6 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Hasil akhir dari perhitungan akan didapatkan matriks :

$$T_0^6 = A_1\, A_2\, A_3\, A_4\, A_5\, A_6 = \begin{bmatrix} n_x & s_x & a_x & p_x \\ n_y & s_y & a_y & p_y \\ n_z & s_z & a_z & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (3.9)$$

Pada Persamaan 3.9 terdapat nilai *n, s* dan *a,* nilai tersebut merupakan titik rotasi, sedangkan, *p* merupakan titik posisi yang dituju *end-effector* dari manipulator robot Denso. Besar nilai variabel *n, s, a,* dan *p* dapat dicari dengan Persamaan yang dihasilkan dari perhitungan perkalian matrik A_1 sampai A_6.

$$\begin{aligned} n_x = &- c6 * (s5 * (c1 * c2 * s3 + c1 * c3 * s2) + c5 * (s1 * s4 \\ &- c4 * (c1 * c2 * c3 - c1 * s2 * s3))) - s6 * (c4 * s1 \\ &+ s4 * (c1 * c2 * c3 - c1 * s2 * s3)) \end{aligned} \quad (3.10)$$

$$\begin{aligned} n_y = &\ s6 * \big(c1 * c4 - s4 * (c2 * c3 * s1 - s1 * s2 * s3)\big) \\ &- c6 * (s5 * (c2 * s1 * s3 + c3 * s1 * s2) - c5 * (c1 * s4 \\ &+ c4 * (c2 * c3 * s1 - s1 * s2 * s3))) \end{aligned} \quad (3.11)$$

$$\begin{aligned} n_z = &\ c6 * (s5 * (c2 * c3 - s2 * s3) + c4 * c5 * (c2 * s3 \\ &+ c3 * s2)) - s4 * s6 * (c2 * s3 + c3 * s2) \end{aligned} \quad (3.12)$$

$$\begin{aligned} s_x = &\ s6 * (s5 * (c1 * c2 * s3 + c1 * c3 * s2) + c5 * (s1 * s4 \\ &- c4 * (c1 * c2 * c3 - c1 * s2 * s3))) - c6 * (c4 * s1 \\ &+ s4 * (c1 * c2 * c3 - c1 * s2 * s3)) \end{aligned} \quad (3.13)$$

$$\begin{aligned} s_y = &\ s6 * (s5 * (c2 * s1 * s3 + c3 * s1 * s2) - c5 * (c1 * s4 \\ &+ c4 * (c2 * c3 * s1 - s1 * s2 * s3))) + c6 * (c1 * c4 \\ &- s4 * (c2 * c3 * s1 - s1 * s2 * s3)) \end{aligned} \quad (3.14)$$

$$\begin{aligned} s_z = &- s6 * (s5 * (c2 * c3 - s2 * s3) + c4 * c5 * (c2 * s3 \\ &+ c3 * s2)) - c6 * s4 * (c2 * s3 + c3 * s2) \end{aligned} \quad (3.15)$$

$$a_x = s5 * \big(s1 * s4 - c4 * (c1 * c2 * c3 - c1 * s2 * s3)\big) - c5 * (c1 * c2 * s3 + c1 * c3 * s2) \quad (3.16)$$

$$a_y = -c5 * (c2 * s1 * s3 + c3 * s1 * s2) - s5 * (c1 * s4 + c4 * (c2 * c3 * s1 - s1 * s2 * s3)) \quad (3.17)$$

$$a_z = c5 * (c2 * c3 - s2 * s3) - c4 * s5 * (c2 * s3 + c3 * s2) \quad (3.18)$$

$$p_x = a2 * c1 * c2 - d4 * (c1 * c2 * s3 + c1 * c3 * s2) - d6 * (c5 * (c1 * c2 * s3 + c1 * c3 * s2) - s5 * (s1 * s4 - c4 * (c1 * c2 * c3 - c1 * s2 * s3))) \quad (3.19)$$

$$p_y = a2 * c2 * s1 - d4 * (c2 * s1 * s3 + c3 * s1 * s2) - d6 * (c5 * (c2 * s1 * s3 + c3 * s1 * s2) + s5 * (c1 * s4 + c4 * (c2 * c3 * s1 - s1 * s2 * s3))) \quad (3.20)$$

$$p_z = d1 + a2 * s2 + d4 * (c2 * c3 - s2 * s3) + d6 * (c5 * (c2 * c3 - s2 * s3) - c4 * s5 * (c2 * s3 + c3 * s2)) \quad (3.21)$$

Titik *p* yang merupakan titik posisi yang berada pada sumbu *x*, *y*, *z* dapat dilihat pada Persamaan 3.19 sampai 3.21 yang merupakan posisi yang dituju dari *end-effector* manipulator robot. Persamaan *forward kinematics* akan digunakan untuk menguji hasil sudut yang telah diperoleh dari perhitungan *inverse kinematics* agar nilai posisi akhir sesuai dengan posisi yang diinginkan.

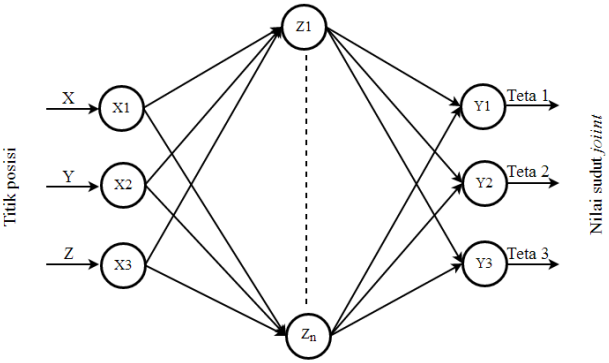
3.4 Desain *Inverse Kinematics* dengan *Neural Network*

Dalam mendesain *inverse kinematics neural network* pertama dilakukan pemilihan jenis struktur dan aktivasi yang digunakan, kedua melakukan perhitungan *feedforward* yang mengolah masukan berupa posisi menjadi keluaran sudut, ketiga melakukan perhitungan *backward* untuk memperkecil *error* output yang terjadi.

3.4.1 Menentukan Struktur *Neural Network* [10]

Dalam tugas akhir ini desain *inverse kinematics neural network* menggunakan struktur *multilayer perceptron*, dimana struktur *neural*

network terdiri dari *input layer, hidden layer*, dan *output layer*. Struktur *inverse kinematics neural network* dapat dilihat pada Gambar 3.7.



Gambar 3.7 Struktur Dasar *Inverse Kinematics Neural Network*

Inverse kinematics neural network terdiri dari satu *input layer* yang berisikan tiga buah *neuron* (x_1, x_2, x_3) yang berfungsi sebagai masukan posisi *x,y,z* dari manipulator robot, lalu satu *hidden layer* yang berisikan sejumlah *neuron* (z_1,..., z_n), dimana jumlah *neuron* dicari dari hasil terbaik penelitian yang dibahas pada bab iv dan menggunakan aktivasi *sigmoid biner* , dan satu lagi *output layer* (y_1, y_2, y_3) yangberisikan tiga buah *neuron* dengan aktivasi *linier* yang menghasilkan besar sudut untuk *joint* manipulator robot.

3.4.2 Perhitungan *Feedforward*

Perhitungan *feedforward* , dimana masukan posisi yang berada pada *input layer inverse kinematics neural network* dikumpulkan dan diaktivasi pada *hidden layer*, lalu setelah itu hasil aktivasi *hidden layer* disatukan di dalam *output layer* dan diaktivasi sehingga mengeluarkan sebuah data berupa besar sudut *joint* manipulator robot. Langkah untuk melakukan perhitungan *feedforward* adalah sebagai berikut :

Langkah 1 : Pemberian nilai awal 0 pada pembobot

Langkah 2 :Masing-masing unit *input* (x_i , i = 1,....n) menerima sinyal *input* x_i berupa nilai posisi x,y,z (*cartesian space*) dan sinyal tersebut disebarkan ke tiap *neuron hidden layer*.

Langkah 3 : Masing-masing unit dilapisan tersembunyi dikalikan dengan faktor pembobot dan dijumlahkan:

$$z\_in_j = \sum_{i=1}^{n} x_i v_{ij} \qquad (3.22)$$

Kemudian menghitung sesuai dengan fungsi aktivasi *sigmoid biner*:

$$z_j = f(z\_in_j)$$

$$z_j = \frac{1}{1+e^{-(z\_in_j)}} \quad (3.23)$$

Langkah 4 : Masing-masing unit *output* (y_k, k=l,2,3,..m) dikalikan dengan faktor pembobot dan dijumlahkan:

$$y\_in_k = \sum_{j=1}^{p} z_j w_{jk} \quad (3.24)$$

hitung kembali dengan fungsi aktivasi *linier*

$$y_k = f(y_{in_k})$$

$$y_k = y\_in_k \quad (3.25)$$

3.4.3 Perhitungan *Backward* (*Backporpagation*)

Setelah melakukan perhitungan *feedforward* , maka akan muncul data yang berupa besar sudut *joint* manipulator robot, dimana besar sudut *joint* yang dihasilkan *inverse kinematics neural network* belum tentu sesuai dengan target tujuan, oleh karena itu dilakukan perhitungan *backpropagation* dimana perhitungan ini mencari eror melakukan revisi bobot pada *inverse kinematics neural network* dan melakukan perhitungan *feedforward* lagi sampai menemukan eror terkecil. Langkah yang harus dilakukan adalah sebagai berikut:

Langkah 5: Masing-masing unit *output* (y_k, k=l,...,m) yang berupa besar sudut $\theta_1, \theta_2, \theta_3$ serta nilai $\theta_4, \theta_5, \theta_6$ dinolkan dimasukkan ke dalm Persamaan *forward kinematics* menjadi posisi keluaran *neural network* x_n, y_n, z_n

Langkah 6: Mencari *gradien error joint* 1 sampai *joint 3*

$$\delta_k = (t_k - y_k) \quad (3.26)$$

Dimana untuk ,

Formulasi target *tetha* 1 sampai *tetha* 3 yang diharapkan :

$$t_1 = \left(tan2^{-1}(y, x)\right) * 180/pi \quad (3.27)$$

$$t_2 = \sqrt{x^2 + y^2} \quad (3.28)$$

$$t_3 = z \quad (3.29)$$

Formulasi target *tetha* 1 sampai *tetha* 3 keluaran :

$$y_1 = \left(tan2^{-1}(y_n, x_n)\right) * 180/pi \quad (3.30)$$

$$y_2 = \sqrt{x_n^2 + y_n^2} \quad (3.31)$$

$$y_3 = z_n \quad (3.32)$$

Menghitung perbaikan faktor pembobot (kemudian untuk memperbaiki w_{jk}):

$$\Delta w_{jk} = \alpha \delta_k z_j \quad (3.33)$$

dan menggunakan nilainya pada semua unit lapisan sebelumnya.

Langkah 7 : Masing-masing pembobot yang menghubungkan unit-unit lapisan *output* dengan unit-unit pada lapisan tersembunyi (z_j,j=1,.,p) dikalikan delta dan dijumlahkan sebagai *input* ke unit-unit lapisan berikutnya.

$$\delta\_in_j = \sum_{k=1}^{m} \delta_k w_{jk} \quad (3.34)$$

Selanjutnya dikalikan dengan turunan dari fungsi aktivasinya untuk menghitung *error*nya.

$$\delta_j = \delta\_in_j * z_j * \left(1 - z_j\right) \quad (3.35)$$

Kemudian menghitung perbaikan pembobot (digunakan untuk memperbaiki v_{ij}.

$$\Delta v_{ij} = \alpha \delta_j x_i \quad (3.36)$$

Langkah 8: Masing-masing *output* unit (yk, k=1,..,m) diperbaiki pembo-botnya (j=0,...,p)

$$w_{jk}(baru) = w_{jk}(lama) + \Delta w_{jk} \quad (3.37)$$

masing-masing unit tersembunyi (z_j, j: 1,....,p) diperbaiki pembobotnya (j=0,...,n).

$$v_{ij}(baru) = v_{ij}(lama) + \Delta v_{ij} \quad (3.38)$$

Langkah 9: Pengulangan langkah 3 sampai dengan langkah 8 sehingga *gradien error* mendekati 0 atau sampai ketelitian yang diinginkan.

3.4.4 Membuat Data Uji untuk *Neural Network*

Data uji dihasilkan dari besar sendi yang dipilih secara acak dan dimasukkan ke dalam Persamaan *forward kinematics* lalu didapatkan beberapa posisi x,y,z. Posisi x,y,z yang didapatkan digunakan sebagai masukan *inverse kinematics neural network* dan hasilnya akan disesuaikan dengan besar sudut sendi dari posisi x,y,z tersebut.

3.4.5 Menentukan Titik – Titik Target Tujuan

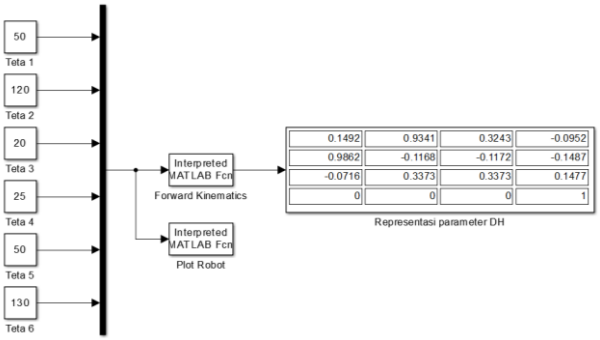
Setelah menguji *inverse kinematics neural network* diberi kan data target posisi yang akan dituju oleh manipulator robot. Titik – titik posisi tersebut dapat digunakan untuk membentuk suatu pola atau Gambar yang tersusun dari sebuah titik koordinat yang disatukan.

3.5 Desain Program Simulasi

Pada perancangan simulasi, *forward kinematics* dan *inverse kinematics neural network* disimulasikan menggunakan *software* Matlab 2014 dengan blok *simulink* . Untuk simulasi perancangan manipulator robot Denso digunakan tambahan *toolbox toolbox* Peter Corke versi 10 untuk menunjang animasi dari manipulator robot.

3.5.1 Simulasi *Forward Kinematics*

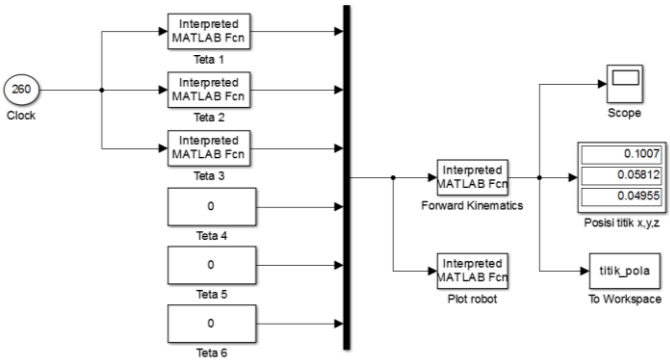
Simulasi *forward kinematics* digunakan untuk menampilkan gerakan manipulator robot yang dihasilkan dari besar nilai sendi yang dimasukkan pada manipulator robot. Pada pergerakan yang dihasilkan, ditampilkan juga parameter DH yang terbentuk dari besar nilai sendi yang dimasukkan. Simulasi ini bertujuan untuk mengetahui posisi *end-effector* manipulator robot dari masukkan berupa besar nilai setiap sendi.



Gambar 3.8 Desain Simulink *Forward Kinematics*

3.5.2 Simulasi *Inverse Kinematics* dengan *Neural Network*

Hasil dari perancangan *inverse kinematics neural network* akan menghasilkan besar nilai sudut pada masing-masing *joint* dengan masukan berupa posisi target x,y,z yang dituju oleh *end-effector* manipulator robot Denso. Titik posisi pada sumbu x,y,z yang dihasilkan oleh simulasi harus sesuai dengan masukan posisi target x,y,z yang diinginkan. Hasil rancangan desain simulasi *inverse kinematic neural network* dapat dilihat pada Gambar 3.10.



Gambar 3.9 Desain Simulink *Inverse Kinematics Neural Network*

--- Halaman ini sengaja dikosongkan --

BAB 4
SIMULASI DAN ANALISA DATA

Pada Bab ini dibahas mengenai pengujian *forward kinematics* sampai solusi *inverse kinematics* dengan *neural network*. *Forward kinematics* menghasilkan *output* berupa titik posisi pada sumbu *x,y,z*. Titik posisi untuk masukan solusi *inverse kinematics neural network* .

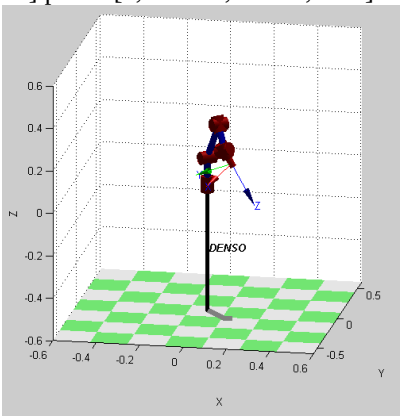
4.1 Pengujian *Forward Kinematics*

Pengujian *forward kinematics* pada Robot Denso dilakukan dengan cara memberikan *input* sudut pada masing-masing *joint* berupa besar sudut dalam satuan derajat. Pemberian *input* sudut pada masing-masing *joint* akan menghasilkan *output* posisi pada *end-effector*. Untuk hasil data keluaran *forward kinematics* dapat dilihat pada Tabel 4.1.

Tabel 4.1 Hasil *Forward Kinematics*

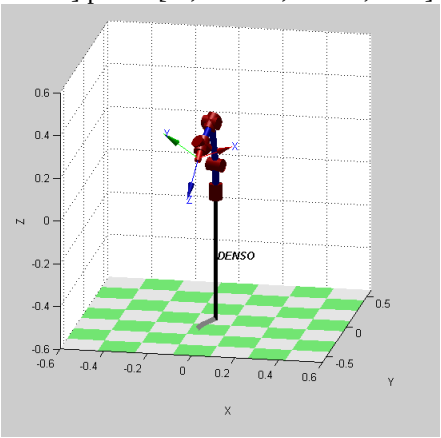
No.	Besar nilai sendi (°)						Posisi		
Uji	θ_1	θ_2	θ_3	θ_4	θ_5	θ_6	x	y	z
1	130	120	35	70	25	0	0,1333	-0,1156	0,1345
2	70	100	35	-75	-10	10	-0,0484	-0,1672	0,1990
3	-100	-10	30	100	-80	20	0,0409	-0,1592	0,2105

Gambar 4.1 menunjukkan Gambaran percobaan pertama dengan sudut [130 120 35 70 25 0] posisi [0,1333 -0,1156 0,1345].



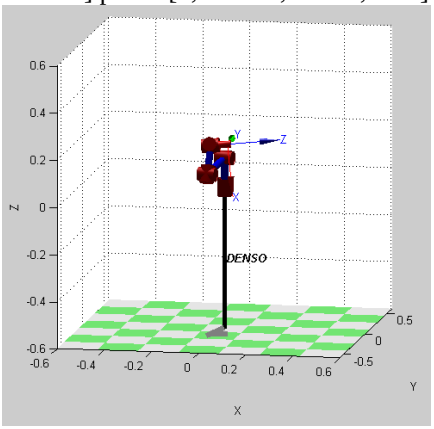
Gambar 4.1 Bentuk Manipulator Robot Percobaan 1

Gambar 4.2 menunjukkan Gambaran percobaan kedua dengan sudut [70 100 35 -75 -10 10] posisi [-0,0484 -0,1672 0,1990].



Gambar 4.2 Bentuk Manipulator Robot Percobaan 2

Gambar 4.3 menunjukkan Gambaran percobaan ketiga dengan sudut [-100 -10 30 100 -80 20] posisi [0,0409 -0,1592 0,2105].



Gambar 4.3 Bentuk Manipulator Robot Percobaan 3

Gambar 4.1 sampai Gambar 4.3 merupakan Gambar bentuk robot hasil dari sudut pada percobaan *forward kinematics*.

4.2 Pembuatan Data Uji *Inverse Kinematics Neural Network*

Data uji yang berupa posisi titik x,y,z diperoleh dari *forward kinematics* dengan besar *joint* yang telah diketahui. Besar *joint* sebagai masukkan *Forward kinematics* dan menghasilkan keluaran berupa titik-titik x,y,z. Untuk data titik x,y,z yang digunakan untuk menguji *inverse kinematics neural network* dapat dilihat pada Tabel 4.2.

Tabel 4.2 Data Uji *Inverse Kinematics Neural Network*

Data ke	Besar Sudut (º)						Posisi		
	θ_1	θ_2	θ_3	θ_4	θ_5	θ_6	x	y	z
1	35	35	135	0	0	0	0,1136	0,0795	0,0564
2	40	40	145	0	0	0	0,1232	0,1034	0,0680
3	45	45	150	0	0	0	0,1286	0,1286	0,0844
4	25	25	125	0	0	0	0,0855	0,0399	0,0475
5	20	20	120	0	0	0	0,0695	0,0253	0,0497

4.3 Pengujian Jumlah *Neuron Hiddden Layer*

Dalam menguji *inverse kinematics neural network* data uji yang telah didapatkan dari *forward kinematics yang* berupa titik x,y,z akan digunakan sebagai masukan dari *inverse kinematics neural network*. Selanjutnya *inverse kinematics neural network* akan menghasilkan data sudut *joint* yang akan langsung diubah menjadi posisi x,y,z *end of effector* dari manipulator robot. Pengujian ini dilakukan sebanyak tiga kali dengan jumlah *neuron* pada *hidden layer* yang berbeda – beda. Titik berangkat manipulator robot pada sudut [30 30 130 0 0 0] dengan posisi awal manipulator robot [0,1006 0,0581 0,0496].

4.3.1 Pengujian dengan 100 *Neuron Hidden Layer*

Hasil pengujian dengan 100 buah *neuron* pada *hidden layer* dapat dilihat pada Tabel 4.3 sampai Tabel 4.4.

Tabel 4.3 Perbandingan Data 100 *Neuron Hidden Layer*

Data	Posisi yang diharapkan			Posisi keluaran *neural network*		
	x	*y*	*z*	*x*	*y*	*z*
1	0,1136	0,0795	0,0564	0,113596039	0,079497359	0,056366169
2	0,1232	0,1034	0,0680	0,123206532	0,103405437	0,068048658
3	0,1286	0,1286	0,0844	0,1285771	0,128577206	0,084439485
4	0,0855	0,0399	0,0475	0,085495255	0,039897757	0,047497816
5	0,0695	0,0253	0,0497	0,069505164	0,02530182	0,049700245

Tabel 4.5 memberikan *Error* antara posisi target dengan keluaran *inverse kinematics neural network*.

Tabel 4.4 *Error* Posisi 100 *Neuron Hidden Layer*

Data ke	*Error* Posisi (m)			*Error* Jarak (m)
	x	y	z	
1	3,96e-06	2,64e-06	3,38e-05	3,42e-05
2	-6,53e-06	-5,44e-06	-4,87e-05	4,94e-05
3	2,29e-05	2,28e-05	-3,95e-05	5,10e-05
4	4,74e-06	2,24e-06	2,18e-06	5,68e-06
5	-5,16e-06	-1,82e-06	-2,45e-07	5,48e-06
RMSE	**1,12e-05**	**1,06e-05**	**3,19e-05**	**3,54e-05**

Hasil pengujian yang tertulis pada Tabel 4.4 dan Tabel 4.5 menggunakan ketelitian 0,00005 dan *learning rate* 0,00025, 100 buah *neuron hidden layer* diselesaikan dengan iterasi rata – rata 42263 selama 139,4 detik.

4.3.2 Pengujian dengan 150 *Neuron Hidden Layer*

Hasil pengujian dengan 150 buah *neuron* pada *hidden layer* dapat dilihat pada Tabel 4.5 sampai Tabel 4.6.

Tabel 4.5 Perbandingan Data 150 *Neuron Hidden Layer*

Data	Posisi yang diharapkan			Posisi keluaran *neural network*		
	x	y	z	x	y	z
1	0,1136	0,0795	0,0564	0,113572947	0,079481076	0,056391393
2	0,1232	0,1034	0,0680	0,123223116	0,103419326	0,067975773
3	0,1286	0,1286	0,0844	0,128569481	0,128569665	0,084432542
4	0,0855	0,0399	0,0475	0,085503239	0,039901507	0,047496003
5	0,0695	0,0253	0,0497	0,069505609	0,025301979	0,049702821

Tabel 4.6 memberikan *error* antara posisi target dengan keluaran *inverse kinematics neural network*.

Tabel 4.6 *Error* Posisi 150 *Neuron Hidden Layer*

Data ke	*Error* posisi (m)			*Error* jarak (m)
	x	y	z	
1	2,71e-05	1,89e-05	8,61e-06	3,41e-05
2	-2,31e-05	-1,93e-05	2,42e-05	3,87e-05
3	3,05e-05	3,03e-05	-3,25e-05	5,40e-05
4	-3,24e-06	-1,51e-06	4,00e-06	5,36e-06
5	-5,61e-06	-1,98e-06	-2,82e-06	6,58e-06
RMSE	**2,12e-05**	**1,82e-05**	**1,87e-05**	**3,36e-05**

Hasil pengujian yang tertulis pada Tabel 4.5 dan Tabel 4.6 menggunakan ketelitian 0,00005 dan *learning rate* 0,0002, 150 buah *neuron hidden layer* diselesaikan dengan iterasi rata – rata 31831 , 113,5 detik.

4.3.3 Pengujian dengan 200 *Neuron Hidden Layer*

Hasil pengujian dengan 200 buah *neuron* pada *hidden layer* dapat dilihat pada Tabel 4.7 sampai Tabel 4.8.

Tabel 4.7 Perbandingan Data 200 *Neuron Hidden Layer*

Data	Posisi yang diharapkan			Posisi keluaran *neural network*		
	x	*y*	*z*	*x*	*y*	*z*
1	0,1136	0,0795	0,0564	0,113587758	0,079491485	0,056402157
2	0,1232	0,1034	0,0680	0,123181776	0,103384541	0,068015522
3	0,1286	0,1286	0,0844	0,128583196	0,128583173	0,084370134
4	0,0855	0,0399	0,0475	0,085458096	0,039880522	0,0474845
5	0,0695	0,0253	0,0497	0,06950488	0,025301806	0,049709171

Tabel 4.7 memberikan *error* antara posisi target dengan keluaran *inverse kinematics neural network*.

Tabel 4.8 *Error* Posisi 200 *Neuron Hidden Layer*

Data ke	*Error* posisi (m)			*Error* jarak (m)
	x	*y*	*z*	
1	1,22e-05	8,52e-06	-2,16e-06	1,51e-05
2	1,82e-05	1,55e-05	-1,55e-05	2,85e-05
3	1,68e-05	1,68e-05	2,99e-05	3,82e-05
4	4,19e-05	1,95e-05	1,55e-05	4,87e-05
5	-4,88e-06	-1,81e-06	-9,17e-06	1,05e-05
RMSE	**2,26e-05**	**1,40e-05**	**1,71e-05**	**3,16e-05**

Hasil pengujian yang tertulis pada Tabel 4.7 dan Tabel 4.8 menggunakan ketelitian 0,00005 dan *learning rate* 0,0002, 200 buah *neuron hidden layer* diselesaikan dengan iterasi rata – rata 26614 selama 95,5 detik.

4.4 Pengujian *Learning Rate Inverse Kinematics Neural Network*

Pengujian bertujuan untuk mencari *learning rate* yang dapat mempercepat iterasi dengan eror yang mendekati ketelitian 0,00005, Dengan 150 buah *neuron* pada *hidden layer*, akan diuji dengan *learning rate* 0,00025 dan 0,0001,

4.4.1 Pengujian dengan *Learning Rate* 0,00025

Hasil pengujian *learning rate* 0,00025 dapat dilihat pada Tabel 4.9 dan 4.10.

Tabel 4.9 Perbandingan Data *Learning Rate* 0,00025

Data	Posisi yang diharapkan			Posisi keluaran *neural network*		
	x	*y*	*z*	*x*	*y*	*z*
1	0,1136	0,0795	0,0564	0,113595435	0,079496904	0,056393552
2	0,1232	0,1034	0,0680	0,123179046	0,103382563	0,067989479
3	0,1286	0,1286	0,0844	0,12862506	0,128624868	0,084368252
4	0,0855	0,0399	0,0475	0,085469421	0,039885657	0,047500779
5	0,0695	0,0253	0,0497	0,069494382	0,025297903	0,04970368

Tabel 4.10 memberikan *error* antara posisi target dengan keluaran *inverse kinematics neural network*.

Tabel 4.10 *Error* Posisi *Learning Rate* 0,00025

Data ke	*Error* posisi (m)			*Error* jarak (m)
	x	*y*	*z*	
1	4,56e-06	3,10e-06	6,45e-06	8,49e-06
2	2,10e-05	1,74e-05	1,05e-05	2,92e-05
3	-2,51e-05	-2,49e-05	3,17e-05	4,75e-05
4	3,06e-05	1,43e-05	-7,79e-07	3,38e-05
5	5,62e-06	2,10e-06	-3,68e-06	7,04e-06
RMSE	**2,03e-05**	**1,51e-05**	**1,53e-05**	**2,96e-05**

Hasil pengujian yang tertulis pada Tabel 4.9 dan Tabel 4.10 menggunakan ketelitian 0,00005 dan *learning rate* 0,00025, 150 buah *neuron hidden layer* diselesaikan dengan iterasi rata – rata 26727 selama 181,2 detik.

4.4.2 Pengujian dengan *Learning Rate* 0,0001

Hasil pengujian *learning rate* 0,0001 dapat dilihat pada Tabel 4.11 dan 4.13.

Tabel 4.11 Perbandingan Data *Learning Rate* 0,0001

Data	Posisi yang diharapkan			Posisi keluaran *neural network*		
	x	*y*	*z*	*x*	*y*	*z*
1	0,1136	0,0795	0,0564	0,113636333	0,079525528	0,056391819
2	0,1232	0,1034	0,0680	0,123188884	0,103390573	0,067966636
3	0,1286	0,1286	0,0844	0,12857423	0,128574198	0,084357984
4	0,0855	0,0399	0,0475	0,08549756	0,039898877	0,047496295
5	0,0695	0,0253	0,0497	0,069512954	0,02530465	0,04969675

Tabel 4.12 memberikan *error* antara posisi target dengan keluaran *inverse kinematics neural network*.

Tabel 4.12 *Error* Posisi *Learning Rate* 0,0001

Data ke	*Error* posisi (m)			*Error* jarak (m)
	x	*y*	*z*	
1	-3,63e-05	-2,55e-05	8,18e-06	4,52e-05
2	1,11e-05	9,43e-06	3,34e-05	3,64e-05
3	2,58e-05	2,58e-05	4,20e-05	5,56e-05
4	2,44e-06	1,12e-06	3,71e-06	4,58e-06
5	-1,30e-05	-4,65e-06	3,25e-06	1,41e-05
RMSE	**2,14e-05**	**1,69e-05**	**2,44e-05**	**3,66e-05**

Hasil pengujian yang tertulis pada Tabel 4.11 dan Tabel 4.12 menggunakan ketelitian 0,00005 dan *learning rate* 0,0002, 150 buah *neuron hidden layer* diselesaikan dengan iterasi rata – rata 34190 selama 216,2 detik.

4.5 Pengujian Ketelitian *Inverse Kinematics Neural Network*

Pengujian ketelitian digunakan untuk mencari pengaruh ketelitian pada proses pencarian solusi *inverse kinematics neural network*. Dengan 150 buah *neuron* pada *hidden layer*, dan *learning rate* 0,00025, akan dilakukan pengujian ketelitian dengan ketelitian 0,0005 dan 0,000005.

4.5.1 Pengujian dengan Ketelitian 0,0005

Hasil pengujian ketelitian 0,0005 terdapat pada Tabel 4.13 dan 4.14.

Tabel 4.13 Perbandingan Data Ketelitian 0,0005

Data	Posisi yang diharapkan			Posisi keluaran *neural network*		
	x	*y*	*z*	*x*	*y*	*z*
1	0,1136	0,0795	0,0564	0,11400682	0,079783448	0,055976635
2	0,1232	0,1034	0,0680	0,123464916	0,103621151	0,068474339
3	0,1286	0,1286	0,0844	0,128245972	0,128247258	0,084793664
4	0,0855	0,0399	0,0475	0,085823432	0,040050047	0,047195469
5	0,0695	0,0253	0,0497	0,069952731	0,02546469	0,049702133

Tabel 4.14 memberikan *error* antara posisi target dengan keluaran *inverse kinematics neural network*.

Tabel 4.14 *Error* Posisi Ketelitian 0,0005

Data ke	*Error* posisi (m)			*Error* jarak (m)
	x	*y*	*z*	
1	-4,07e-04	-2,83e-04	4,23e-04	6,52e-04
2	-2,65e-04	-2,21e-04	-4,74e-04	5,87e-04
3	3,54e-04	3,53e-04	-3,94e-04	6,36e-04
4	-3,23e-04	-1,50e-04	3,05e-04	4,69e-04
5	-4,53e-04	-1,65e-04	-2,13e-06	4,82e-04
RMSE	**3,66e-04**	**2,46e-04**	**3,61e-04**	**5,70e-04**

Hasil pengujian yang tertulis pada Tabel 4.13 dan Tabel 4.14 menggunakan ketelitian 0,00005 dan *learning rate* 0,00025, 150 buah *neuron hidden layer* diselesaikan dengan iterasi rata – rata 18817 selama 174,2 detik.

4.5.2 Pengujian dengan Ketelitian 0,000005

Hasil uji ketelitian 0,000005 terdapat pada Tabel 4.15 dan 4.16.

Tabel 4.15 Perbandingan Data Ketelitian 0,000005

Data	Posisi yang diharapkan			Posisi keluaran *neural network*		
	x	*y*	*z*	*x*	*y*	*z*
1	0,1136	0,0795	0,0564	0,113598893	0,079499212	0,056399404
2	0,1232	0,1034	0,0680	0,123202464	0,103402057	0,068000942
3	0,1286	0,1286	0,0844	0,128471855	0,128472838	0,084422913
4	0,0855	0,0399	0,0475	0,085497065	0,039898621	0,047497047
5	0,0695	0,0253	0,0497	0,069497728	0,025299172	0,04969993

Tabel 4.16 memberikan *error* antara posisi target dengan keluaran *inverse kinematics neural network*.

Tabel 4.16 *Error* Posisi Ketelitian 0,000005

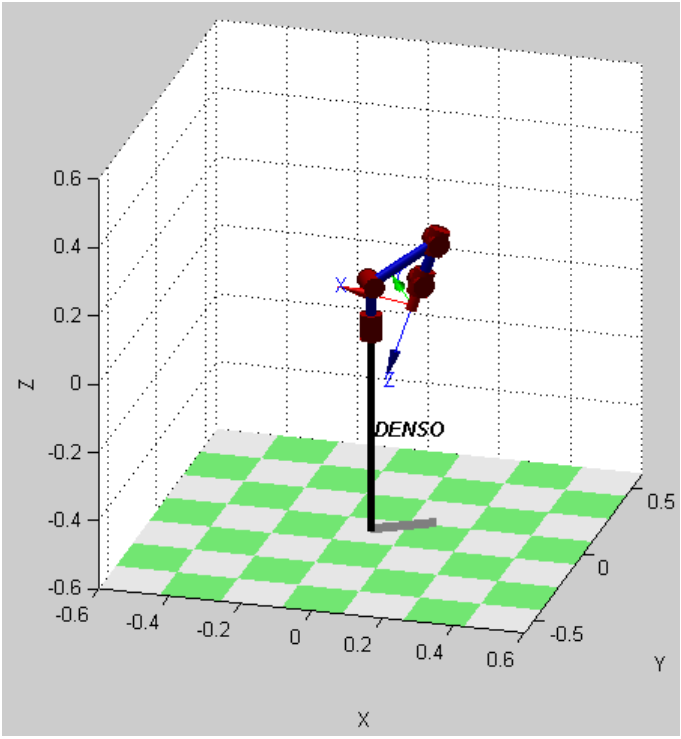
Data ke	*Error* posisi (m)			*Error* jarak (m)
	x	*y*	*z*	
1	1,11e-06	7,88e-07	5,96e-07	1,48e-06
2	-2,46e-06	-2,06e-06	-9,42e-07	3,35e-06
3	1,28e-04	1,27e-04	-2,29e-05	1,82e-04
4	2,94e-06	1,38e-06	2,95e-06	4,39e-06
5	2,27e-06	8,28e-07	7,00e-08	2,42e-06
RMSE	**5,73e-05**	**5,69e-05**	**1,03e-05**	**8,14e-05**

Hasil pengujian yang tertulis pada Tabel 4.15 dan Tabel 4.16 menggunakan ketelitian 0,000005 dan *learning rate* 0,00025, 150 buah

neuron hidden layer diselesaikan dengan iterasi rata – rata 298866 selama 2757,4 detik.

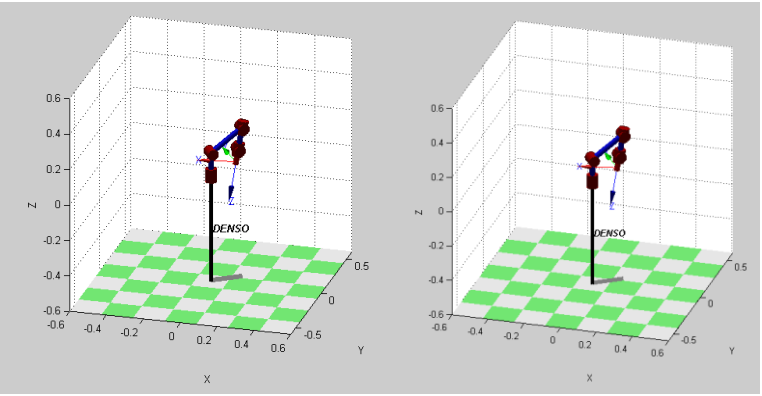
4.6 Plot Hasil Data Pengujian Data ke 1 Sampai Data ke 5

Hasil data pengujian 1 sampai 5 ditunjukkan pada Gambar 4.4 – 4.14 Untuk titik berangkat robot adalah dari sudut [30 30 130 0 0 0] dan berada pada posisi titik [0,1006 0,0581 0,0496], dapat dilihat pada Gambar 4.4.



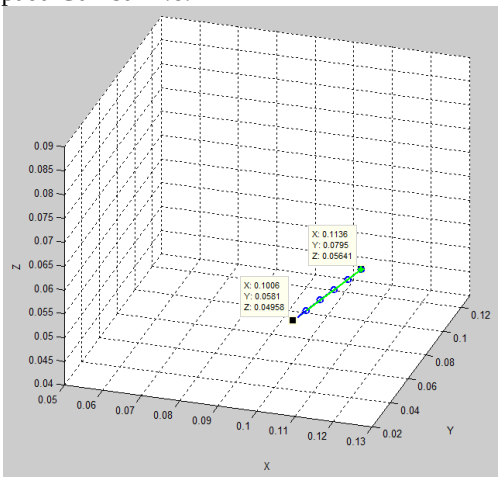
Gambar 4.4 Keadaan Awal Posisi Manipulator Robot

Untuk bentuk manipulator robot pengujian data ke 1, titik posisi tujuan [0,1136 0,0795 0,0564], dapat dilihat pada Gambar 4.5 , dimana sebelah kiri adalah posisi yang diharapkan dan sebelah kanan adalah posisi keluaran *neural network*.



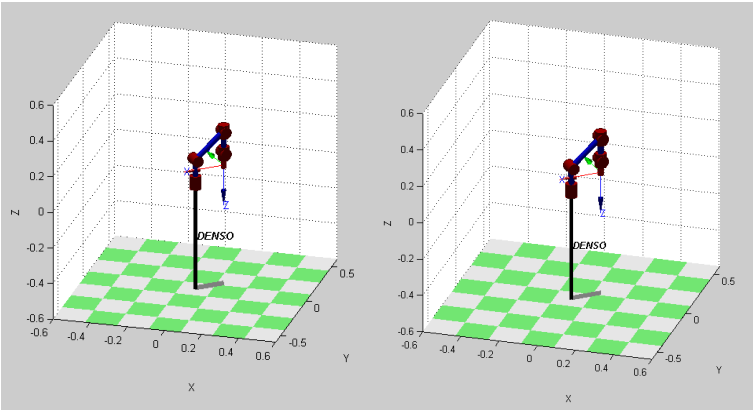
Gambar 4.5 Bentuk Manipulator Robot Pengujian Posisi Data 1

Sedangkan untuk plot titik posisi harapan dan keluaran *neural network* dapat dilihat pada Gambar 4.6.



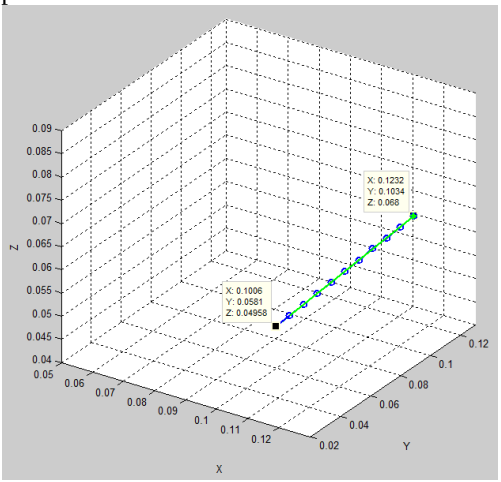
Gambar 4.6 Posisi Titik Awal ke Titik Tujuan Data 1

Untuk bentuk manipulator robot pengujian data ke 2 titik posisi tujuan [0,1232 0,1034 0,0680], dapat dilihat pada Gambar 4.7 , dimana sebelah kiri adalah posisi yang diharapkan dan sebelah kanan adalah posisi keluaran *neural network*.



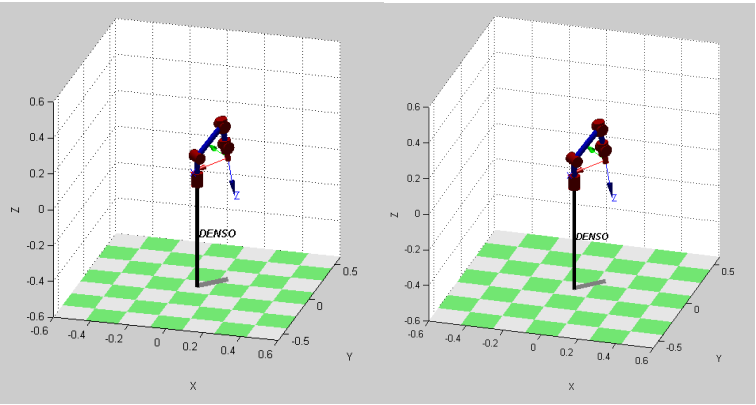
Gambar 4.7 Bentuk Manipulator Robot Pengujian Posisi Data 2

Sedangkan untuk plot titik posisi harapan dan keluaran *neural network* dapat dilihat pada Gambar 4.8.



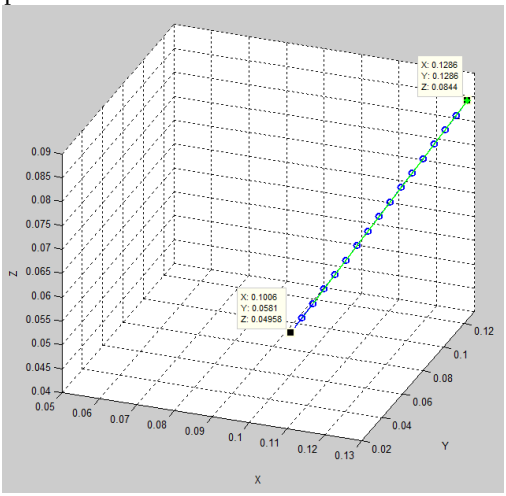
Gambar 4.8 Posisi Titik Awal ke Titik Tujuan Data 2

Untuk bentuk manipulator robot pengujian data ke 3 titik posisi tujuan [0,1286 0,1286 0,0844], dapat dilihat pada Gambar 4.9 , dimana sebelah kiri adalah posisi yang diharapkan dan sebelah kanan adalah posisi keluaran *neural network*.



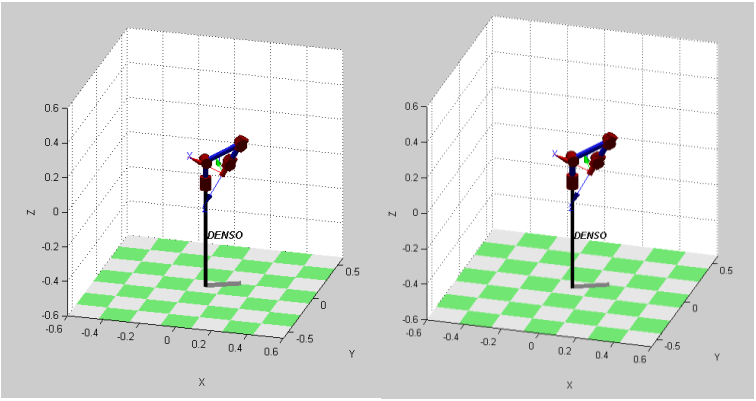
Gambar 4.9 Bentuk Manipulator Robot Pengujian Posisi Data 3

Sedangkan untuk plot titik posisi harapan dan keluaran *neural network* dapat dilihat pada Gambar 4.10.



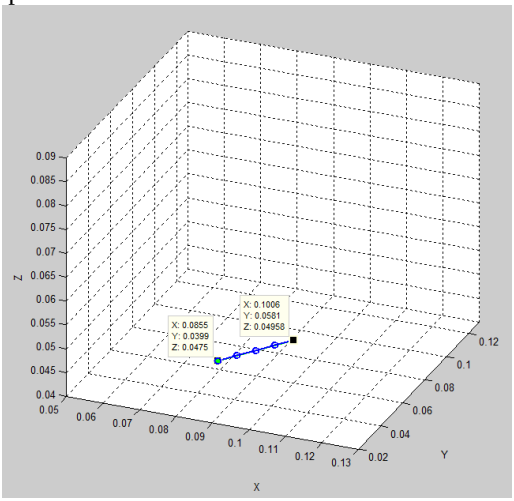
Gambar 4.10 Posisi Titik Awal ke Titik Tujuan Data 3

Untuk bentuk manipulator robot pengujian data ke 4 titik posisi tujuan [0,0855 0,0399 0,0475], dapat dilihat pada Gambar 4.11 , dimana sebelah kiri adalah posisi yang diharapkan dan sebelah kanan adalah posisi keluaran *neural network*.



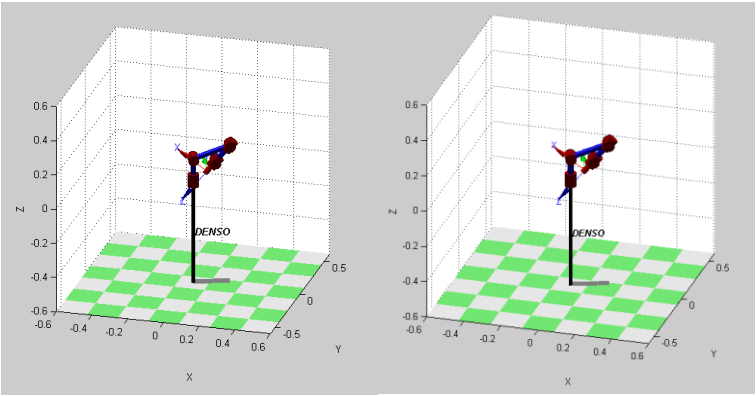
Gambar 4.11 Bentuk Manipulator Robot Pengujian Posisi Data 4

Sedangkan untuk plot titik posisi harapan dan keluaran *neural network* dapat dilihat pada Gambar 4.12.



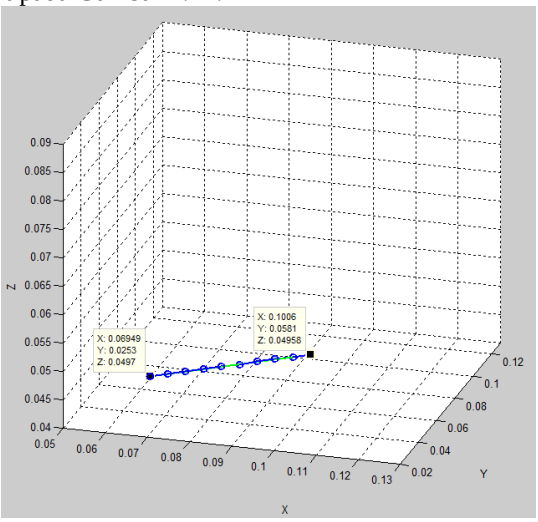
Gambar 4.12 Posisi Titik Awal ke Titik Tujuan Data 4

Untuk bentuk manipulator robot pengujian data ke 5 titik posisi tujuan [0,0695 0,0253 0,0497], dapat dilihat pada Gambar 4.13 , dimana sebelah kiri adalah posisi yang diharapkan dan sebelah kanan adalah posisi keluaran *neural network*.



Gambar 4.13 Bentuk Manipulator Robot Pengujian Posisi Data 5

Sedangkan untuk plot titik posisi harapan dan keluaran *neural network* dapat dilihat pada Gambar 4.14.



Gambar 4.14 Posisi Titik Awal ke Titik Tujuan Data 5

4.7 *Inverse Kinematics Neural Network* untuk Pola Persegi

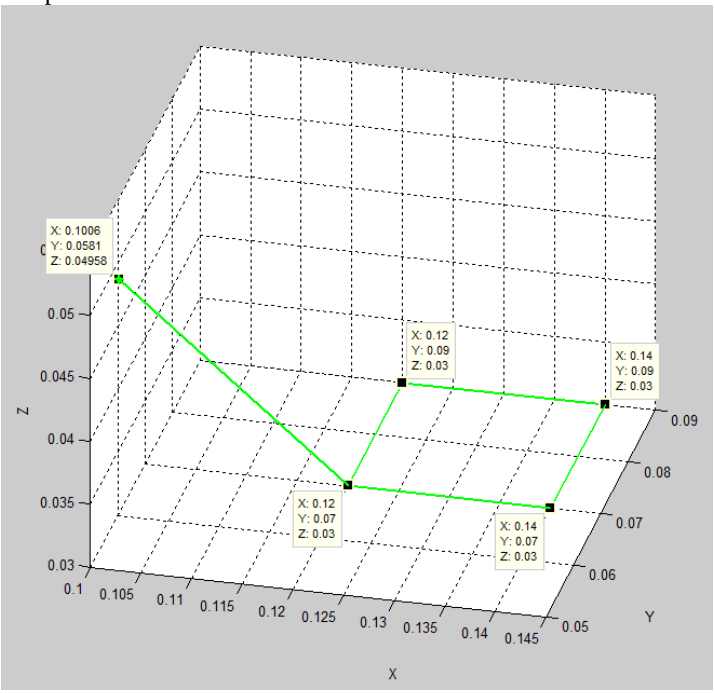
Setelah pengujian menggunakan data uji, *inverse kinematics neural network* dicoba untuk membuat suatu pola persegi dari masukan titik

x,y,z. Untuk menyusun sebuah pola diperlukan titik x,y,z berangkat manipulator robot sampai titik x,y,z sebagai tujuan akhir robot. Dalam Tugas Akhir ini manipulator robot akan membentuk sebuah persegi dengan sudut berangkat [30 30 130 0 0 0] koordinat x,y,z [0,1006 0,0581 0,0496]. Koordinat persegi dapat dilihat pada Tabel 4.18.

Tabel 4.17 Koordinat Target Pola Persegi

Titik	x	y	z
1	0,1200	0,0700	0,0300
2	0,1400	0,0700	0,0300
3	0,1400	0,0900	0,0300
4	0,1200	0,0900	0,0300
5	0,1200	0,0700	0,0300
Akhir	0,1006	0,0581	0,0496

Untuk hasil plot grafik persegi dengan target yang diharapkan dapat dilihat pada Gambar 4.15.



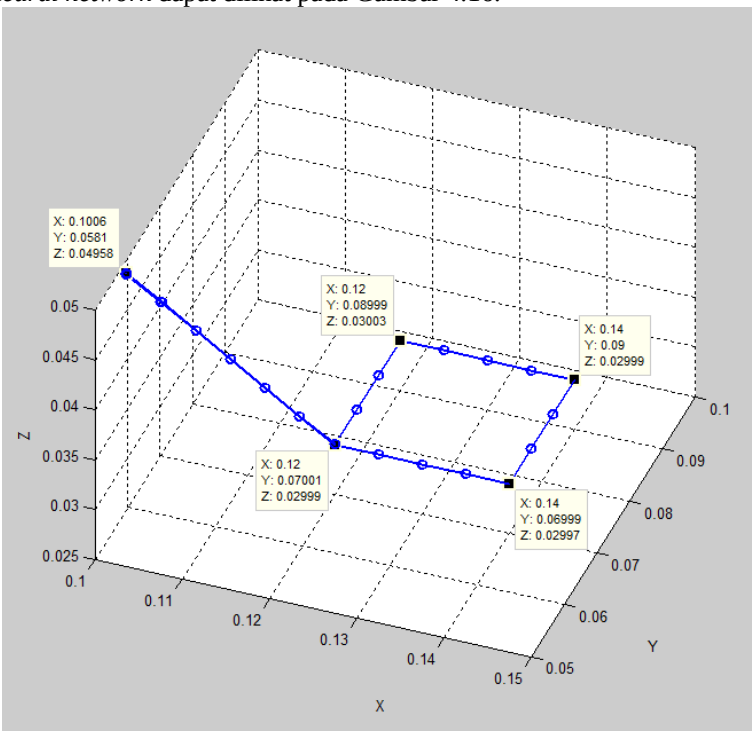
Gambar 4.15 Titik Persegi yang Diharapkan

Tabel 4.18 Koordinat Titik x,y,z Persegi Keluaran *Neural Network*

Titik	x	y	z
1	0,1199984855	0,0699992562	0,0299910712
2	0,1399953399	0,0699978023	0,0299757251
3	0,1400067601	0,0900044925	0,0300222416
4	0,1199687540	0,0899764771	0,0299539516
5	0,1199910624	0,0699946543	0,0300071702
Akhir	0,1006153892	0,0580904074	0,0495363075

Hasil pengujian yang tertulis pada Tabel 4.19 menggunakan ketelitian 0,000005 dan *learning rate* 0,00025, 150 buah *neuron hidden layer* diselesaikan dengan 47396 iterasi , waktu 256,3 detik.

Untuk hasil plot pola persegi keluaran dari *inverse kinematics neural network* dapat dilihat pada Gambar 4.16.

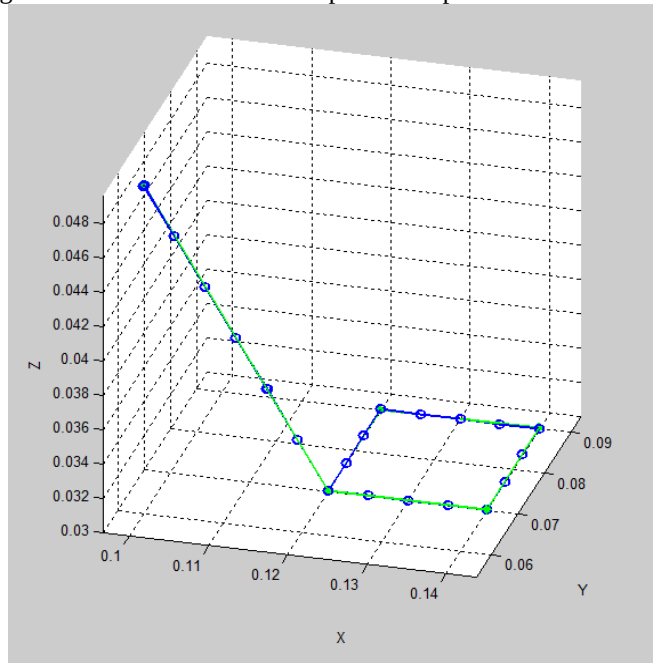


Gambar 4.16 Titik Persegi Keluaran *Neural Network*

Tabel 4.19 *Error* Titik Target dengan Titik *Neural Network*

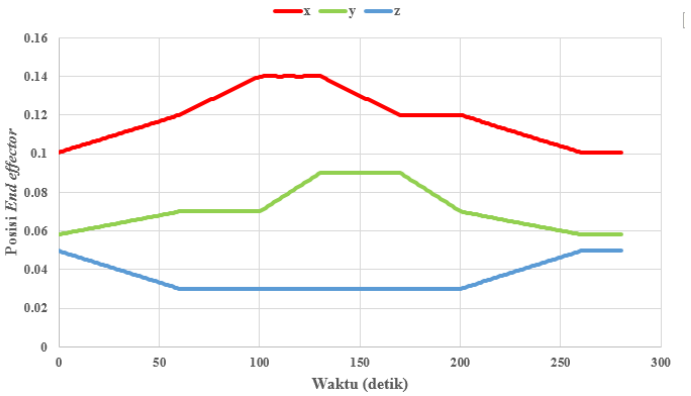
Titik	x (m)	y (m)	z (m)	*Error* Jarak (m)
1	1,51e-06	7,44e-07	8,93e-06	9,09e-06
2	4,66e-06	2,20e-06	2,43e-05	2,48e-05
3	-6,76e-06	-4,49e-06	-2,22e-05	2,37e-05
4	3,12e-05	2,35e-05	4,60e-05	6,04e-05
5	8,94e-06	5,35e-06	-7,17e-06	1,26e-05
Akhir	1,46e-05	8,33e-06	4,27e-05	4,59e-05
RMSE	**1,49e-05**	**1,06e-05**	**2,93e-05**	**3,46e-05**

Untuk grafik perbandingan antara data pola persegi target dengan pola persegi keluaran *Neural Network* dapat dilihat pada Gambar 4.17.



Gambar 4.17 Perbandingan Antara Titik Target dan Keluaran

Grafik hasil pergerakan *end-effector* manipulator robot dari titik berangkat menuju ke titik akhir untuk membentuk suatu pola persegi dapat dilihat pada Gambar 4.17. dimana garis merah merupakan pergerakan x, garis hijau merupakan pergerakan y ,dan garis biru merupakan pergerakan z.



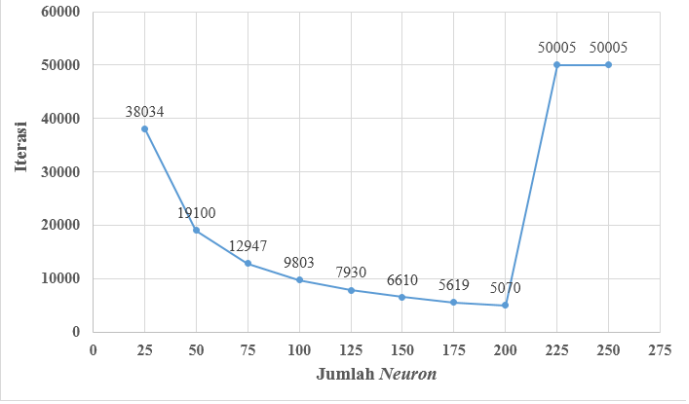
Gambar 4.18 Pergerakan Manipulator Robot Denso

4.8 Grafik Perbandingan

Perbandingan antara jumlah *Neuron* pada *Hidden Layer* dengan Iterasi maupun dengan *Error* Jarak diperlihatkan ke dalam bentuk gambar. Gambar perbandingan dihasilkan dari data yang diletakkan pada halaman lampiran.

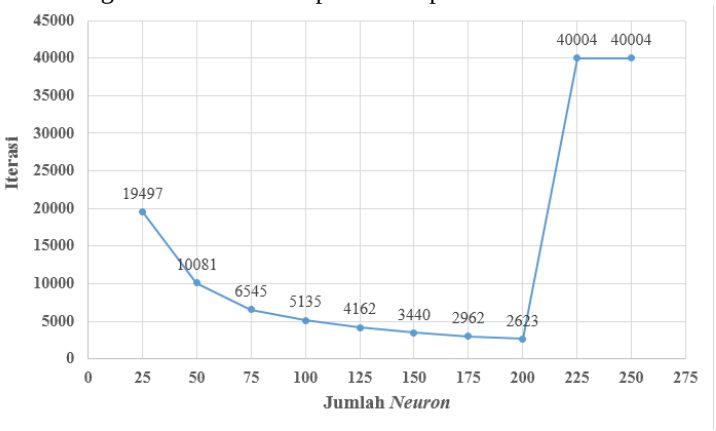
4.8.1 Jumlah *Neuron* dengan Iterasi (2 Target)

Perbandingan antara jumlah *Neuron* dalam *Hidden Layer Neural Network* dengan Iterasi untuk mendapatkan solusi kinematika balik dari titk berangkat menuju titik target data 1 dapat dilihat pada Gambar 4.19.



Gambar 4.19 Perbandingan dengan Iterasi Data 1

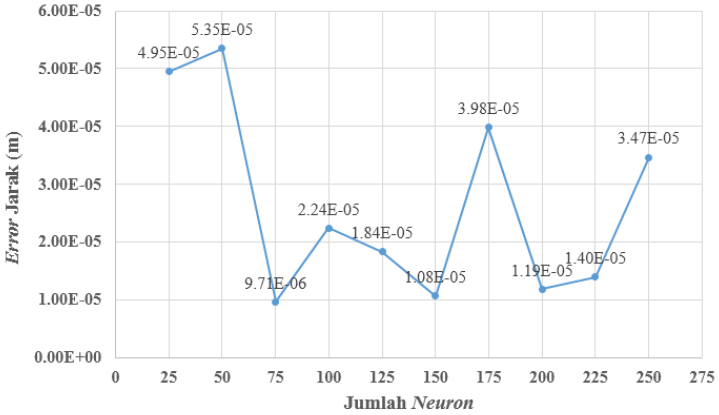
Untuk perbandingan antara jumlah *Neuron* dalam *Hidden Layer Neural Network* dengan Iterasi data 4 dapat dilihat pada Gambar 4.20.



Gambar 4.20 Perbandingan dengan Iterasi Data 4

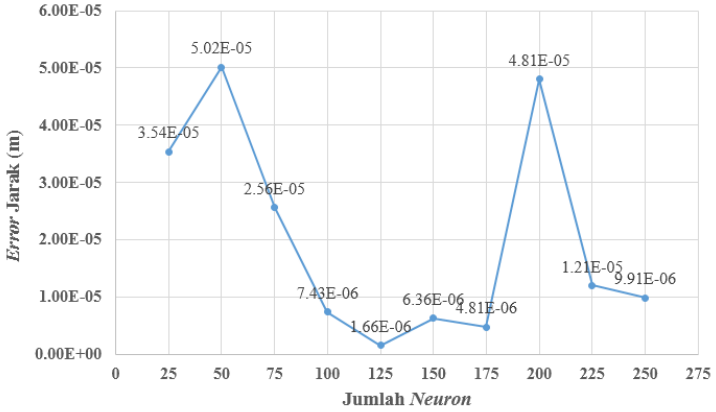
4.8.2 Jumlah *Neuron* dengan *Error* Jarak (2 Target)

Perbandingan antara jumlah *Neuron* dalam *Hidden Layer Neural Network* dengan *Error* Jarak untuk mendapatkan solusi kinematika balik manipulator robot dari titk berangkat menuju titik target dari data 1 dapat dilihat pada Gambar 4.21.



Gambar 4.21 Perbandingan dengan *Error* Jarak Data 1

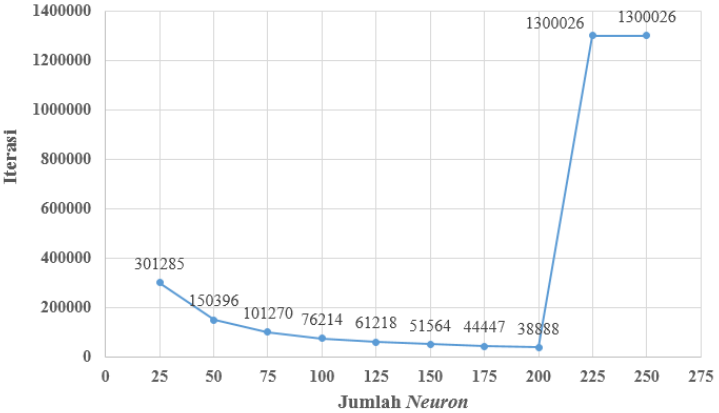
Untuk perbandingan antara jumlah *Neuron* dalam *Hidden Layer Neural Network* dengan *Error* Jarak data 4 dapat dilihat pada Gambar 4.21.



Gambar 4.22 Perbandingan dengan *Error* Jarak Data 4

4.8.3 Jumlah *Neuron* dengan Iterasi (Pola Persegi)

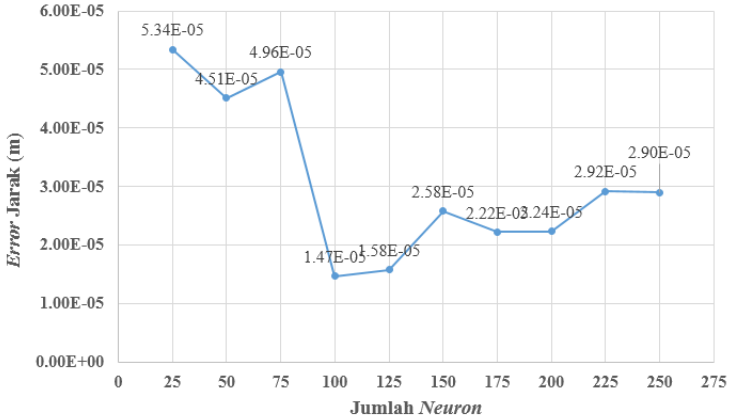
Perbandingan antara jumlah *Neuron* dalam *Hidden Layer Neural Network* dengan Iterasi untuk mendapatkan solusi kinematika balik manipulator robot dari titk berangkat menuju titik target dengan pembentukan pola persegi dapat dilihat pada Gambar 4.23.



Gambar 4.23 Perbandingan dengan Iterasi Pola Persegi

4.8.4 Jumlah *Neuron* dengan *Error* Jarak (Pola Persegi)

Perbandingan antara jumlah *Neuron* dalam *Hidden Layer Neural Network* dengan *Error* Jarak untuk mendapatkan solusi kinematika balik manipulator robot dari titk berangkat menuju titik target dengan pembentukan pola persegi dapat dilihat pada Gambar 4.24.



Gambar 4.24 Perbandingan dengan *Error* Jarak Pola Persegi

--- Halaman ini sengaja dikosongkan --

BAB 5
PENUTUP

5.1 Kesimpulan

Berdasarkan hasil simulasi percobaan pada bab analisa sistem, ada beberapa hal yang dapat disimpulkan dari penelitian pada Tugas Akhir ini, yaitu :

1. Semakin besar jumlah *neuron* , *error* jarak semakin kecil dan iterasi semakin cepat, dibuktikan dengan jumlah *neuron* 100 memiliki *error* jarak 3,54e-05 m dengan iterasi rata-rata 42263, sedangkan dengan jumlah *neuron* 200 memiliki *error* jarak 3,16e-05 m dengan rata-rata 26614.
2. Semakin besar *learning rate* , *error* jarak semakin kecil dan iterasi semakin cepat, dibuktikan dengan *learning rate* 0,00025 memiliki *error* jarak 2,96e-05 m dengan iterasi rata-rata 26727, sedangkan dengan *learning rate* 0,0001 memiliki *error* jarak 3,66e-05 m dengan iterasi rata-rata 34190.
3. Semakin besar ketelitian, iterasi semakin cepat, dibuktikan dengan ketelitian 0,0005 deselesaikan dengan iterasi rata-rata 18817, sedangkan dengan ketelitian 0,000005 memiliki iterasi rata-rata 298866.
4. Manipulator robot Denso mampu menemukan posisi titik untuk pola persegi yang diselesaikan dengan *inverse kinematics neural network* dengan *error* x sebesar 1,49e-05 m, *error* y sebesar 1,06e-05 m, *error* z sebesar 2,93e-05 m.

5.2 Saran

Sebagai pengembangan penelitian Tugas Akhir ini, ada beberapa hal yang dapat digunakan untuk perkembangan selanjutnya, yaitu:

1. Pengembangan simulasi selanjutnya, dengan menghitung orientasi manipulator robot Denso 6-DOF.
2. Disarankan untuk mencoba bentuk pola lain yang terdiri dari garis lengkung.
3. Untuk selanjutnya diharapkan pembelajaran menggunakan pengawasan sehingga solusi *inverse kinematics* dapat diselesaikan sejalan dengan simulasi pergerakan.

--- Halaman ini sengaja dikosongkan --

DAFTAR PUSTAKA

[1] Spong, M. W. & Vidyasagar, M., 1989, *Robot Dynamics and control.* 1st ed. Canada: John Wiley & Sons, Inc.

[2] ....., n.d. *Quanser 6-Axis Articulated Robot.* Canada: Quanser,Inc.

[3] Pradana, S. S., 2015. Solusi Inverse Kinematics Menggunakan Logika Fuzzy pada Robot Manipulator Denso 6-DoF. *Tugas Akhir*. Teknik Elektro. Surabaya: Institut Teknologi Sepuluh Nopember.

[4] Spong, M. W., Hutchinson, S. & Vidyasagar, M., 2004, *Robot Dynamics and Control.* 2nd ed. s.l.:s.n.

[5] Prasetia, I. E., 2015. Inverse Kinematics dengan Solusi Closed Form Pada Robot Denso Manipulator. *Tugas Akhir*. Teknik Elektro. Surabaya: Institut Teknologi Sepuluh Nopember.

[6] Handritoar, 2013, *Kinematik.* [Online] Available at: https://handritoar.wordpress.com/2013/08/28/1-kinematik/ [Accessed 9 May 2017].

[7] Pandjaitan, L. W., 2007, *Dasar-dasar Komputasi Cerdas.* Yogyakarta: ANDI OFFSET.

[8] Ekasari, R. P., 2013. Pengatur Temperatur Pada Heat Exchanger Dengan Menggunakan Metode Jaringan Saraf Tiruan Prediktif. *Tugas Akhir*. Teknik Elektro. Surabaya: Institut Teknologi Sepuluh Nopember.

[9] Ratna Wati, D. A., 2011, *Sistem Kendali Cerdas.* Yogyakarta: GRAHA ILMU.

[10] Duka, A. V., 2013, Neural Network Based Inverse Kinematics Solution for Trajectory Tracking of a Robotic Arm. *Procedia Technology*. Romania. 12 (2014) 20 – 27.

[11] Corke, P., n.d. *Resources for robotics education: code, books and MOOCS.* [Online] Available at: http://petercorke.com/wordpress/toolboxes/robotics-toolbox [Accessed 10 2 2017].

--- Halaman ini sengaja dikosongkan --

LAMPIRAN

1. Program *Forward Kinematics*

```
function [N] = forward1(mas)
global a2 d1 d4 d6
global c1 c2 c3 c4 c5 c6 s1 s2 s3 s4 s5 s6
global nx ny nz sx sy sz ax ay az px py pz
function [N] = forward1(mas)
global a2 d1 d4 d6
global c1 c2 c3 c4 c5 c6 s1 s2 s3 s4 s5 s6
global nx ny nz sx sy sz ax ay az px py pz
a2= 0,21;d1= 0,125;d4= 0,122;d6= 0,070;1=mas(1);
t2=mas(2);t3=mas(3);t4=mas(4);t5=mas(5);
t6=mas(6);
%inisialisasi
c1=cosd(t1);c2=cosd(t2);c3=cosd(t3);c4=cosd(t4);
c5=cosd(t5);c6=cosd(t6);s1=sind(t1);s2=sind(t2);
s3=sind(t3);s4=sind(t4);s5=sind(t5);s6=sind(t6);

%inisialisasi matrix
nx = - c6*(s5*(c1*c2*s3 + c1*c3*s2) + c5*(s1*s4 -
    c4*(c1*c2*c3 - c1*s2*s3))) - s6*(c4*s1 +
    s4*(c1*c2*c3 - c1*s2*s3));
ny =s6*(c1*c4 - s4*(c2*c3*s1 - s1*s2*s3)) -
    c6*(s5*(c2*s1*s3 + c3*s1*s2) - c5*(c1*s4 +
    c4*(c2*c3*s1 - s1*s2*s3)));
nz =c6*(s5*(c2*c3 - s2*s3) + c4*c5*(c2*s3 +
    c3*s2)) - s4*s6*(c2*s3 + c3*s2);
sx =s6*(s5*(c1*c2*s3 + c1*c3*s2) + c5*(s1*s4 -
    c4*(c1*c2*c3 - c1*s2*s3))) - c6*(c4*s1 +
    s4*(c1*c2*c3 - c1*s2*s3));
sy =s6*(s5*(c2*s1*s3 + c3*s1*s2) - c5*(c1*s4 +
    c4*(c2*c3*s1 - s1*s2*s3))) + c6*(c1*c4 -
    s4*(c2*c3*s1 - s1*s2*s3));
sz = - s6*(s5*(c2*c3 - s2*s3) + c4*c5*(c2*s3 +
    c3*s2)) - c6*s4*(c2*s3 + c3*s2);
ax =s5*(s1*s4 - c4*(c1*c2*c3 - c1*s2*s3)) -
    c5*(c1*c2*s3 + c1*c3*s2);
```

```
ay = - c5*(c2*s1*s3 + c3*s1*s2) - s5*(c1*s4 +
    c4*(c2*c3*s1 - s1*s2*s3));
az =c5*(c2*c3 - s2*s3) - c4*s5*(c2*s3 + c3*s2);
px =a2*c1*c2 - d4*(c1*c2*s3 + c1*c3*s2) -
    d6*(c5*(c1*c2*s3 + c1*c3*s2) - s5*(s1*s4 -
    c4*(c1*c2*c3 - c1*s2*s3)));
py =a2*c2*s1 - d4*(c2*s1*s3 + c3*s1*s2) -
    d6*(c5*(c2*s1*s3 + c3*s1*s2) + s5*(c1*s4 +
    c4*(c2*c3*s1 - s1*s2*s3)));
pz =d1 + a2*s2 + d4*(c2*c3 - s2*s3) +
    d6*(c5*(c2*c3 - s2*s3) - c4*s5*(c2*s3 +
    c3*s2));
M=[nx sx ax px;
   ny sy ay py;
   nz sz sz pz
   0 0 0 1]
N=[px py pz];
```

2. Program *Inverse Kinematics Neural Network*

```
clear all;
close all;
clc;
%INIALISASI
tha=[30 30 130 0 0 0];
Mth=[forward11(tha)
    0,0695    0,0253    0,0497];
NH = 150; %jumlah neuron hidden
NO = 3; %jumlah neuron output
a=0;
for ktk=1:length(Mth(:,1))-1
    dxyz=Mth(ktk+1,:)-Mth(ktk,:);
    dtot=sqrt(dxyz(1)^2+dxyz(2)^2+dxyz(3)^2);
    d=0,005;
    jdpoin=floor(dtot/d);
    dp(1)=dxyz(1)/jdpoin; %dx
    dp(2)=dxyz(2)/jdpoin; %dy
    dp(3)=dxyz(3)/jdpoin; %dz
    % Perhitungan bobot random
    bv = 0*rand(3,NH);%bobot layer hidden
    bw = 0*rand(NH,3); %bobot layer output
```

```
for p=1:jdpoin
    xyztar=zeros(p,3);
    xyztar(p,:)= Mth(ktk,:)+
                 p*[dp(1) dp(2) dp(3)];
    gradw(1)=10;
    gradw(2)=10;
    gradw(3)=10;
    I(p)=0;
    Y = zeros(size(xyztar(p,:)));
    %mengembalikan nilai 0 per iterasi
    while (abs(gradw(1))>0,00005 ||
          abs(gradw(2))>0,00005 ||
          abs(gradw(3))>0,00005)
        %Feedforward
        for j=1:NH
            sumxv(j)=0;
            for i=1:3 %jumlah neuron hidden
                sumxv(j) =  sumxv(j) +
                            (dp(i)*bv(i,j));
            end
            z_in(j) = sumxv(j);
            Z(j)=1/(1+exp(-z_in(j)));
        End

        for k=1:NO
            sumvw(k)=0;
            for j=1:NH
                sumvw(k)=   sumvw(k) +
                            (Z(j)*bw(j,k));
            end
            y_in(k)=sumvw(k);
            Y(p,k)=y_in(k);
        end
        thad= tha+
              [Y(p,1) Y(p,2) Y(p,3) 0 0 0];
        if thad(1)>160
            thad=thad-[320 0 0 0 0 0];
        elseif thad(1)<-160
            thad=thad+[320 0 0 0 0 0];
```

```
else
    thad=thad;
end
if thad(2)>50
    thad=thad-[0 30 0 0 0 0];
elseif thad(2)<20
    thad=thad+[0 30 0 0 0 0];
else
    thad=thad;
end
if thad(3)>160
    thad=thad-[0 0 40 0 0 0];
elseif thad(3)<120
    thad=thad+[0 0 40 0 0 0];
else
    thad=thad;
end
xyzn=forward11(thad);
sd1(p)=atan2(xyztar(p,2),
xyztar(p,1))*180/pi;
sd2(p)=sqrt((xyztar(p,1)^2)+
(xyztar(p,2)^2));
sd3(p)=xyztar(p,3);

sd1n(p)=atan2(xyzn(2),
xyzn(1))*180/pi;
sd2n(p)=sqrt((xyzn(1)^2)+
(xyzn(2)^2));
sd3n(p)=xyzn(3);

%Backpropagation
gradw(1)=sd1(p)-sd1n(p);
gradw(2)=sd2(p)-sd2n(p);
gradw(3)=sd3(p)-sd3n(p);

%Inialisasi update bobot
tnd2=-1;
if sign(gradw(2))>0
    tnd2=1;
end
```

```
tnd3=1;
if sign(gradw(3))>0
    tnd3=-1;
end
clc;
I(p)=I(p)+1;
I(p)
I;
[sd1(p) sd1n(p) gradw(1)
    sd2(p) sd2n(p) gradw(2)
    sd3(p) sd3n(p) gradw(3)
    thad(:,1:3)
    Mth(ktk,:)
    xyztar(p,:)
    xyzn]
[jdpoin p ktk]
ljpemw(1)=0,00025;
ljpemw(2)=0,00025;
ljpemw(3)=0,00025;
ljpemv=0,1;
if abs(gradw(1))<0,005
    ljpemw(1)=0,0000001+rand(1,1)*
              (0,0001-0,0000001);
end
if abs(gradw(2))<0,005
    ljpemw(2)=0,0000001+rand(1,1)*
              (0,0001-0,0000001);
end
if abs(gradw(3))<0,005
    ljpemw(3)=0,0000001+rand(1,1)*
              (0,0001-0,0000001);
end
if I(p) > 5000
    ljpemv=0,000001;
end
if I(p) > 200000
    break
end
```

```
%Hitung delta bobot w
for j=1:NH
    delw(j,1)=ljpemw(1)*
              sign(gradw(1))*Z(j);
    delw(j,2)=ljpemw(2)*tnd2*Z(j);
    delw(j,3)=ljpemw(3)*tnd3*Z(j);
end

for j=1:NH
    grad_in(j)=0;
    for k=1:NO
        grad_in(j)=grad_in(j)+
                 (gradw(k)*bw(j,k));
    end
end

for j=1:NH
    gradh(j)=grad_in(j)*
             Z(j)*(1-Z(j));
end
%Hitung delta bobot v
for i=1:3
    for j=1:NH
        delv(i,j)=ljpemv*
                  gradh(j)*dp(i);
    end
end

%Bobot w
for j=1:NH
    for k=1:NO
        bw(j,k)=bw(j,k)+delw(j,k);
    end
end
%Bobot v
for i=1:3
    for j=1:NH
        bv(i,j)=bv(i,j)+delv(i,j);
    end
end
```

```
        end
        hasil(p,:)=thad;
    end
    hasilakh(a+1:a+p,:)=hasil(1:p,:);
    a=a+p;
end
```

3. Data Grafik Perbandingan Data 1

Jumlah *Neuron*	Data	*Error* Jarak (m)	Iterasi	Waktu (detik)
25	1	4,95e-05	38034	86,64999157
50	1	5,35e-05	19100	51,752116
75	1	9,71e-06	12947	41,08374568
100	1	2,24e-05	9803	35,01736345
125	1	1,84e-05	7930	32,03516674
150	1	1,08e-05	6610	29,0345341
175	1	3,98e-05	5619	26,6753088
200	1	1,19e-05	5070	25,48451614
225	1	1,40e-05	50005	265,5073021
250	1	3,47e-05	50005	276,6388416

4. Data Grafik Perbandingan Data 4

Jumlah *Neuron*	Data	*Error* Jarak (m)	Iterasi	Waktu (detik)
25	4	3,54e-05	19497	45,59691745
50	4	5,02e-05	10081	30,80062604
75	4	2,56e-05	6545	20,83395937
100	4	7,43e-06	5135	18,6888691
125	4	1,66e-06	4162	16,64510934
150	4	6,36e-06	3440	15,34008636
175	4	4,81e-06	2962	14,22697837
200	4	4,81e-05	2623	13,31325827
225	4	1,21e-05	40004	206,4750031
250	4	9,91e-06	40004	213,2050853

5. Data Grafik Perbandingan Pola Persegi

Jumlah *Neuron*	*Error* Posisi (m)			*Error* Jarak (m)	Iterasi	Waktu (detik)
	x	y	z			
25	2,71e-05	1,68e-05	4,29e-05	5,34e-05	301285	710,4285
50	2,58e-05	1,64e-05	3,32e-05	4,51e-05	150396	387,2822
75	2,80e-05	1,66e-05	3,74e-05	4,96e-05	101270	332,3632
100	9,52e-06	6,07e-06	9,48e-06	1,47e-05	76214	273,4227
125	9,59e-06	5,76e-06	1,11e-05	1,58e-05	61218	264,3663
150	1,63e-05	9,25e-06	1,78e-05	2,58e-05	51564	202,8686
175	1,51e-05	9,03e-06	1,36e-05	2,22e-05	44447	178,2127
200	1,22e-05	7.82e-06	1,71e-05	2,24e-05	38888	196,8389
225	1,80e-05	1,80e-05	1,43e-05	2,92e-05	1300026	8030,312
250	2.36e-05	1.06e-05	1.32e-05	2.90e-05	1300026	47497,1

RIWAYAT PENULIS

Tegar Wangi Arlean, dilahirkan di Trenggalek pada tanggal 24 Maret 1994, Penulis merupakan anak pertama dari tiga bersaudara. Penulis telah menempuh pendidikan formal di SDN 2 Gemaharjo (2000-2006), SMPN 1 Trenggalek (2006-2009), SMAN 1 Trenggalek (2009-2012) dan D3 Jurusan Teknik Elektro dengan bidang studi *Computer Control* (2012-2015). Selanjutnya terdaftar di Jurusan Teknik Elektro Institut Teknologi Sepuluh Nopember Surabaya. Di Jurusan Teknik Elektor penulis mengambil bidang studi Teknik Sistem Pengaturan dengan judul Tugas akhir "KINEMATIKA BALIK MANIPULATOR ROBOT DENSO DENGAN METODE *NEURAL NETWORK*".

Contact Person :
Email : tegar.wangi@gmail.com
No.HP : 087755012400
Line : tegarwangi

--- Halaman ini sengaja dikosongkan --