**T.C.**
**SELÇUK ÜNİVERSİTESİ**
**FEN BİLİMLERİ ENSTİTÜSÜ**

# ÜÇ SERBESTLİK DERECELİ BİR ROBOTUN, YAPAY SİNİR AĞLARI VE GENETİK ALGORİTMA KULLANILARAK ENGELLİ ORTAMDA ÇARPIŞMASIZ YÖRÜNGE PLANLAMASI

**Serhat AKSUNGUR**

**YÜKSEK LİSANS TEZİ**
**MAKİNE MÜHENDİSLİĞİ ANABİLİM DALI**
**KONYA, 2009**

**ÖZET**

**Yüksek Lisans Tezi**

**ÜÇ SERBESTLİK DERECELİ BİR ROBOTUN, YAPAY SİNİR AĞLARI VE GENETİK ALGORİTMA KULLANILARAK ENGELLİ ORTAMDA ÇARPIŞMASIZ YÖRÜNGE PLANLAMASI**

**Serhat AKSUNGUR**

**Selçuk Üniversitesi Fen Bilimleri Enstitüsü**

**Makine Mühendisliği Anabilim Dalı**

**Danışman : Yrd. Doç. Dr. Koray KAVLAK**

**2009, 102 Sayfa**

**Jüri : Prof. Dr. Ziya ŞAKA**

**Yrd. Doç. Dr. Koray KAVLAK**

**Yrd. Doç. Dr. İlhan ASİLTÜRK**

Bu çalışmada, iki dönel ve bir lineer mafsala sahip üç serbestlik dereceli SCARA tipi bir robotun ters kinematik analizi ve yörünge planlaması yapılmıştır. Robotun çalışma alanına engel yerleştirilerek hareket grafiği çizilmiş, çarpışma olup olmadığı gözlenmiş ve robot kolunun dönme yönü belirlenmiştir. Hedef ve engel koordinatları rastgele seçilen iki bin adet örnek oluşturularak bu işlem her örneğe uygulanmıştır. Bu işlemlerin sonucunda elde edilen hedef ve engel koordinat değerleri yapay sinir ağı (YSA) için giriş, hesaplanan mafsal açı değerleri de çıkış seti olarak belirlenmiştir. Eğitim için çok katmanlı geri yayılım ağı ve aktivasyon fonksiyonu olarak da sigmoid fonksiyonu kullanılmıştır. Geri yayılım ağı, Genetik Algoritma ile optimizasyon yöntemini kullanarak ağırlık ve momentum değerlerini güncellemiştir. Kabul edilebilir hata değerine ulaşılarak ağın eğitimi tamamlanmıştır. İstenilen ve YSA ile hesaplanan değerler için karşılaştırma grafiği çizilerek sonuçların uygun olduğu gözlenmiştir.



**Anahtar Kelimeler:** Yapay sinir ağları, genetik algoritma, ters kinematik analiz, yörünge planlaması.

## ABSTRACT

**Master's Thesis**

**REALIZE THE COLLISION FREE PATH PLANNING WITH OBSTACLES ENVIRONMENT OF THREE DEGREES OF FREEDOM ROBOT USING ARTIFICAL NEURAL NETWORKS AND GENETIC ALGORITHM**

**Serhat AKSUNGUR**
**Selcuk University**
**Graduate School of Natural and Applied Sciences**
**Department of Mechanical Engineering**
**Supervisor: Assist. Prof. Dr. Koray KAVLAK**
**2009, 102 Pages**
**Jury : Prof. Dr. Ziya ŞAKA**
**Assist. Prof. Dr. Koray KAVLAK**
**Assist. Prof. Dr. İlhan ASİLTÜRK**

In this study, the inverse kinematics and path-planning for three degrees of freedom SCARA robot which has two rotary and one linear joint are carried out. The motion graphic is formed after an obstacle put on the robot's workspace, condition of collision is observed and direction of rotation of robot arm is determined. Two thousand exemplars are constituted with randomly selected goal and obstacle coordinates and this procedure is applied to all of this exemplars. As a result of this procedure, obtained goal and obstacle coordinate values are given to artificial neural network (ANN) as input and calculated joint angle values are defined as output. For training, a multilayer back propagation network and sigmoid function as a transfer function are used. Back propagation network updates the weight and momentum values with genetic algorithm optimization method. After acceptable error value was attained training network has completed. The comparison graphic was drawn for desired values and values calculated by ANN. The results presented here were observed to be valid.



**Key Words:** Artificial neural networks, genetic algorithm, inverse kinematics, path planning.

# ÖNSÖZ

Toplumların gelişmişlik düzeyini mevcut teknolojilerinin belirlediği yüzyılımızda, üretimin sürekliliği, çeşitliliği ve kalitesi ön plana çıkmıştır. Bu sebeple insan gücü yerini otomasyonlara ve karmaşık üretim teknolojilerine bırakmıştır.

Ürün çeşitliliğinin yarattığı sistem değişiklikleri, mevcut sistem modernizasyonlarında ciddi zaman kayıplarına yol açmaktadır.

Otomasyonların yerine geçebilecek ve her geçen gün daha da geliştirilen düşünebilen robotlar, geleceğimizin ütopyası değil gerçeği olmak zorundadır.

Bu tez çalışmamda, Yapay Sinir Ağları ve Genetik Algoritma teknikleri ile üç serbestlik dereceli bir manipülatör üzerinde çalışarak öğrenebilen bir sistem meydana getirdim.

Çalışmalarımda bana yardımcı olan danışman hocam Sayın **Yrd. Doç. Dr. Koray KAVLAK**' a, bu tez konusunu seçerken beni destekleyen hocam Sayın **Yrd. Doç. Dr. Arif ANKARALI**' ya ve bu tezi tamamlayabilmem için verdiği sonsuz manevi destek için aileme teşekkürü bir borç bilirim.

Serhat AKSUNGUR

## İÇİNDEKİLER

## SİMGELER

$q(t)$ : Mafsallara ait açılar vektörü

$a_i$ : Mafsal eksenleri arasındaki ortak normalleri boyunca ölçülen en kısa mesafe

$\alpha_i$ : Mafsal eksenleri arasındaki $a_i$' ye dik bir düzlem üzerinde ölçülen açı

$d_i$ : Uzuv açıklığı

$\theta_i$ : Eklem açısı

L : Eklem boyu

${}^{x}_{y}T$ : Robotun x elemanının y elemanına göre konum ve oryantasyonunu gösteren transformasyon matrisi

$\theta(t_i)$ : Uç işlevcisinin $t_i$ anındaki konumu

$\dot{\theta}(t_i)$ : Uç işlevcisinin $t_i$ anındaki hızı

$\ddot{\theta}(t_i)$ : Uç işlevcisinin $t_i$ anındaki ivmesi

$s_i$ : Yörünge planlaması için yazılan fonksiyonun katsayıları

$X_i$ : Yapay sinir ağları için, i'inci giriş değeri

$W_{ij}$ : Yapay sinir ağları için, j'inci elemandan i'inci elemana bağlantı ağırlığı

$q_i$ : Eşik (threshold) değeri

$F_h$ : Gizli katmandaki her bir nöron için aktifleşme

$F_j$ : Çıkış katmanındaki her bir nöron için aktifleşme

$\delta_j$ : Her çıkış nöronu için geriye yayılma hatası

$\delta_h$ : Gizli katmanlardan en son katmandaki nöronlar için geriye yayılma hatası

$TH$ : Toplam hata

$\alpha$ : Momentum katsayısı

## KISALTMALAR

| | |
|---|---|
| ABD | : Amerika Birleşik Devletleri |
| ADALINE | : Adaptive Linear Neuron *(Adaptif Doğrusal Nöron)* |
| ART | : Adaptive Resonance Theory *(Adaptif Rezonans Teorisi)* |
| BP | : Backpropagation *(geri yayılım)* |
| EDBD | : Extended Delta – Bar – Delta *(Genişletilmiş Delta – Bar – Delta)* |
| FELC | : Feedback Error Learning Control *(Hata Geri Beslemeli Öğrenme Kontrolü)* |
| GA | : Genetic Algorithm *(Genetik Algoritma)* |
| İE | : İşlemci Eleman |
| İEÇ | : İşlem Eleman Çıkış Değeri |
| LVQ | : Linear Vector Quantization *(Doğrusal Vektör Parçalama)* |
| MADALINE | : Multiple Adaline *( Çoklu ADALINE)* |
| MLP | : Multi Layer Perseptron *(Çok Katmanlı Algılayıcı)* |
| PID | : Proportional Integral Derivative control *(Oransal İntegral Türevsel kontrol)* |
| SCARA | : Selectively Compliant Articulated Robot Arm *(Seçici Serbest Esnemeli Robot Kolu)* |
| SOM | : Self Organizing Map *(Düzenleyici Harita)* |
| VGM | : Visibility Graph Method *(Önceden Görünebilirlik Grafik Metodu)* |
| VLSI | : Very Large Scale Integration *(Büyük Ölçekli Entegre Devre)* |
| YSA | : Yapay Sinir Ağları |

## ŞEKİLLER DİZİNİ

## TABLOLAR DİZİNİ

# 1. GİRİŞ

Otomasyonlar, üretim teknolojilerinin yeni işçileridir. Yatırım maliyetlerinin yüksek olmasına rağmen insan faktörünün ortadan kalkması ile ürün kalitesindeki artış ve işçilik masraflarındaki düşüş sayesinde amortisman süreleri çok düşüktür.

Ancak ürün çeşitliliğinin artması, bu mevcut sistemler üzerinde sürekli revizyonlar yapılmasını zorunlu hale getirmiştir. Yapılan bu revizyonlar esnasında üretimin yavaşlaması/durması sebebi ile işletmelerin süreklilikleri sekteye uğramaktadır. Ayrıca işletme üzerine ek maliyetler yüklemektedir.

Yeniden programlanabilir sistemlerin bu tarz problemlerine karşın düşünebilen ve karar verebilen sistemler, yapılması gereken değişikliklere kendileri karar verebilirler ve daha önce karşılaşmadıkları farklı problemler için uygun çözümler üretebilirler.

Ülkemizde otomasyon teknolojisine yakın zamanda geçilmiş olması ve henüz tam olarak rayına oturmamış olması, daha ileri teknolojileri hayata geçirebilmek açısından önemli bir engel olarak karşımıza çıkmaktadır. Ancak bilim ve teknolojinin çok hızlı geliştiği günümüzde yapay zekânın yakın zamanda hayatımıza gireceği ve otomasyonların yavaş yavaş yerlerini uzman sistemlere devredeceği yadsınamaz bir gerçektir.

Bu çalışmada robotik sistemler için önemli problemlerden biri olan yörünge planlamasının yapay zekâ yardımıyla robot tarafından oluşturulması sağlanmaya çalışılmıştır. Karar verme özelliği sayesinde yapılacak değişikliklerde kaybolan zamanın büyük oranda azalacağı gösterilmeye çalışılmıştır.

# 2. KAYNAK ARAŞTIRMASI

Pashkevich ve ark. (2006), yaptıkları çalışmada, bir kaynak robotunun çarpışmasız yörünge planlaması için yapay sinir ağları (YSA) yaklaşımı konusunu incelemişlerdir. Öncelikle harmonik fonksiyonlar kullanılarak yörünge planlaması yapılmış, ardından YSA kullanılarak harmonik fonksiyonlar için bir model oluşturulmuştur. Geliştirilen yörünge planlaması algoritması, kaynak robot birimlerinin karşılaştıkları düzensiz engelleri de hesaba katmaktadır ve çarpışma miktarı indirgeme kontrolü sağlamaktadır.

Noguchi ve Terao (1997), bu çalışmada tarım alanlarında kullanılan mobil robotlar için YSA ve genetik algoritma (GA) kullanarak yörünge planlaması yapmışlardır. YSA' nın lineer olmayan karmaşık sistemler üzerinde çözüme kolay ulaşabilmesi ve genetik algoritmaların, karmaşık yapıların optimizasyonunda iyi sonuç vermesi özelliklerinden faydalanmışlardır.

Tiryaki ve Kazan (1995), çalışmalarında üç eklemli bir SCARA robotunu ele almış ve dinamiğini YSA ile modellemişlerdir. Robotun Lagrange-Euler dinamik denklemleri çıkarılmıştır. Modellenen SCARA robot kolu, iki dönme ve bir öteleme hareketi yapmaktadır. Verilen zaman aralıklarındaki konum, hız ve ivme değerleri YSA' ya giriş, tork değerleri ise çıkış olarak kullanılmıştır. Bu çalışmada çok katmanlı ileri beslemeli geri yayılım ağı ve transfer fonksiyonu olarak da logaritmik sigmoid fonksiyonu kullanılmıştır.

Shibata ve ark. (1997), altı serbestlik dereceli bir robot için üç boyutlu çalışma alanında, hareket planlama kısıtlandırma metodu önermişlerdir. Bu metotla el açılarını optimize etmek için genetik algoritma kullanmışlardır. Uygun bir fonksiyon için deneyimli bir operatörü referans alarak, değer fonksiyonunu tanımlamışlardır. Önerilen metotta fazla olan parametreler hesaba katılmadan yol saptandığından operatörün iş gücü azaltılmıştır.

Kert (2006), çalışmasında, tek kamera ile alınan sayısal görüntülere, görüntü işleme tekniklerini uygulayarak, dairesel kesitli nesneler ile bu nesneler arasındaki engellerin konumlarının bulunmasını ve nesnelere erişim sırasının optimizasyonunu gerçekleştirmiştir. Görüntü işleme sonucu elde edilen nesnelerin üç boyutlu koordinatlarını, soruna özgü geliştirilen genetik algoritma programına aktararak, robot kolun nesnelere erişim sırasının optimizasyonunu sağlamıştır.

Nearchou (1998), yaptığı çalışmada, geliştirilmiş GA kullanarak karmaşık ortamlarda çalışan robotların ters kinematik analiz probleminin çözümü ile uğraşmıştır. Araştırmacının amacı, hem elin konumsal hatalarını hem de robot eklemlerinin yer değiştirme hareketlerini minimize etmektir. Geliştirilmiş GA algoritmasının etkinliğini farklı robotlar üzerinde birçok deney yaparak gözlemlemiş, sunulan çözümün Pseudoinverse ve temel GA metotlarıyla elde edilen çözümlerle karşılaştırmasını yapmış ve önerilen algoritmanın belirgin bir şekilde daha iyi olduğu sonucuna ulaşmıştır.

Laribi ve ark. (2007), tavsiye edilen çalışma alanı için DELTA robotunun analizi ve boyutsal sentezi üzerinde çalışmışlardır.

Toogood ve ark. (1995), genetik algoritma kullanarak, üç serbestlik dereceli dönel bir robotun özelleştirilmiş başlangıç ve bitiş konfigürasyonları ile, engellerin bilindiği bir ortamda çarpışmasız hareket ettirilmesi üzerinde çalışmışlardır. Çarpışmadan kaçınmak ve belirlenen mafsal kısıtlarının içinde kalmak koşuluyla yörünge, minimum mesafe, zaman, mafsal torkları veya bunların kombinasyonu için optimize edilmiştir.

Aydın ve Temeltaş (2004), çalışmalarında mobil robotlarda, önceden görünebilirlik grafik metodu olarak adlandırılan VGM (Visibility Graph Method) ile zaman-optimal yörünge üzerinde durmuşlardır. Yörünge üzerinde olası tüm eğri

kısımlar kümesi içinden diferansiyel evrim metodu ile engellerle çakışmayan yörüngenin seçilmesini gerçekleştirmişlerdir.

Çonkur (2003), gereğinden çok serbestlik dereceli robot kolları için potansiyel alan metodunu kullanarak yörünge planlaması yapan bir yazılım üzerinde çalışmıştır. Bu yazılımın en önemli özellikleri, engellerin ve robotların ekrana çizilmesi, potansiyel alanın iki ve üç boyutlu görüntülerinin elde edilmesi ve robotun hedefe varmasının gözlemlenebilmesidir.

Khoogar ve Parker (1999), 3 serbestlik dereceli bir manipülatörün çarpışmasız yörünge planlaması için genetik algoritmaların kullanımı üzerinde çalışmışlardır. İki örnek karşılaştırılmış, uygulamanın avantaj ve dezavantajları tartışılmıştır.

Cherif ve ark. (1995), yapay sinir ağları ile ivmelenme seviyesinde ters kinematik analiz üzerinde çalışmışlardır. Fazlalık problemi performans fonksiyonu minimize edilerek çözülmüştür. Bu metotla az sayıda iterasyonla kesin sonuç elde etmişlerdir.

Er ve Liev (1997), bu çalışmada önceden belirlenmiş eklem uzayında hareket yörüngesinin, yapay sinir ağı tabanlı kontrolör tarafından öğrenilmesi üzerinde çalışmışlardır. Yörünge izi, yörünge üzerindeki varyasyonlar, farklı ağırlık değerleri, Hata Geri Beslemeli Öğrenme Kontrolü (Feedback Error Learning Control – FELC) ile öğrenilmiş ve karşılaştırılmıştır.

Zhang ve ark. (2008), bir mobil robotun üç boyutlu ortamda genetik algoritmalar kullanarak optimum yörünge planlaması üzerinde çalışmışlardır. Robot, ortamda bulunan bazı engellerden sıçrayarak aşamamakta veya inememektedir. Engellerin pozisyonları ve yükseklikleri bilinmekte ve yörünge, genetik algoritma yardımıyla optimize edilmektedir. Yazarlara göre bu çalışma, etkileyiciliği ve uygulanabilirliği ile iki boyutlu çalışmalara göre atak niteliğindedir.

Galicki (2005), engel tarafından oluşturulan ceza fonksiyonu eğimi ve işlem hatasının dahil olduğu transpoze jakobiyen konrolcüsü içeren hesaplanabilir basit bir sınıf önermiştir. Kontrol şemasını elde etmek için Lyapunov kararlılık teorisi kullanılmıştır. Harici ceza fonksiyonu yaklaşımıyla engelle çarpışma olmaması sağlanmıştır.

Blackmore ve Williams (2006), bu çalışmada Disjunctive Programming yöntemi ile optimum çarpışmasız yörüngenin bulunması üzerinde çalışmışlardır. Bu metotla, manipülatörün tüm çalışma alanında, çalışma alanından konfigürasyon alanına, pahalı olan engel haritalama yollarını elimine ederek optimum yörüngeyi bulmaya çalışmışlardır.

Fu ve ark. (1987), tarafından kaleme alınan bu kitapta, robotların hareket için gerekli olan matematiksel hesaplamalarının (kinematik analiz, dinamik analiz, yörünge planlaması, robot manipülatörlerin kontrolü) yanı sıra bilgisayar tabanlı robotik uygulamaları da (algılama, görme, robot programlama dilleri, robot zekası ve görev planlama) detaylı şekilde incelenmiştir.

Mitchell'in (1997), Machine Learning eserinde öğrenme algoritmaları üzerinde durulmuş, örneklerle açıklanmıştır.

Nabiyev'in (2003), Yapay Zekâ eserinde, yapay zekânın geniş tanımı yapılmış, yapay zekânın kullanım alanları, türleri ve yöntemleri üzerinde durulmuştur.

Öztemel (2003), kaleme aldığı Yapay Sinir Ağları isimli eserinde yapay sinir ağlarının geniş tanımını yapmış, öğrenmede kullanılan YSA algoritmalarını sınıflandırmış ve tümünü ayrıntılı olarak açıklamıştır.

Öztürk (2007), çalışmasında SCARA tipi bir manipülatörün yapay sinir ağları ile eğitilmesi üzerinde durmuştur.

Duran (2007), puma tipi bir robotun PID ile kontrolü üzerinde çalışmıştır. Robot için gerekli olan kinematik, dinamik ve yörünge planlaması hesaplamalarını detaylı olarak anlatmıştır.

Şahin (2006), çalışmasında PID kontrolü ile SCARA tipi bir robotun yörünge kontrolünün yapılması üzerinde çalışmıştır. Yörünge planlamasında bilgisayar kontrolünün kullanılması üzerinde durmuştur.

Bingül ve Küçük (2005), yayınladıkları Robot Tekniği I adlı eserlerinde, robotların düz ve ters kinematik analizleri üzerinde durmuş, dinamik analiz ve yörünge planlaması hakkında hesaplamalar sunmuşlardır.

Şen (2004), Genetik Algoritmalar ve En İyileme Yöntemleri isimli eserinde en iyileme yöntemleri hakkında açıklamalarda bulunmuş ve genetik algoritmaların işleyişi hakkında detaylı bilgiler vermiştir.

Haupt (1998), Practical Genetic Algorithms adlı eserinde genetik algoritmalar kullanılarak yapılan örnekler üzerinde durmuştur.

# 3. ROBOTLARIN GENEL ÖZELLİKLERİ

Robot kelimesini ilk olarak Çek asıllı filozof ve oyun yazarı Karel Capek, 1922 yılında “Rossum’ s Universal Robot (R.U.R.) (Rossum’un Evrensel Robotları))” isimli oyunu içerisinde kullanmıştır. Robot kelimesi Çek dilinde “Robotnik” olarak geçmekte ve “işçi” veya “esir” anlamına gelmektedir [24].

Robotlar için “Bir operatörün çeşitli reflekslerini ve zekâsını kullanışının basit bir uygulamasıdır” denilebilir. Robotlar çevreleriyle ve nesnelerle sürekli etkileşim halindedir. Bu etkileşimin sağlanması, görevin önceden belirlenerek robota tanıtılması ile gerçekleştirilir. Günümüzde robotlar, gelişen sanayiye ayak uydurmakta zorlanan işçi sınıfının alternatifleri haline gelmişlerdir. Bunun başlıca üç sebebi vardır [24];

1. Verimliliği arttırma ihtiyacı,
2. Kaliteyi iyileştirme ihtiyacı,
3. Hassas ve tekrarlanabilir işlerin kolaylıkla yapılabilmesi.

Robotlar, çeşitli malzemeleri, parçaları, takımları veya özel aletleri, bir dizi görevin gerçekleştirilmesi için, programlanmış hareketler boyunca taşıyacak şekilde tasarlanmış yeniden programlanabilir manipülatörler olarak da tanımlanabilirler.

Günümüzde robot tanımı üzerinde dünyada kesin olarak bir fikir birliği sağlanamamıştır. ABD ve Avrupa ülkeleri, robot tanımlamalarında, bir robotun üç niteliğe sahip olması gerektiğini belirtmektedirler. Bu nitelikler;

1. Manipülasyon yapma, yani cisimlerin yerini değiştirme ve üzerlerinde işlem yapma,
2. Programlanabilirlik, yani benzer türdeki, çeşitli işlemleri yapabilme olanağı,
3. Algılama sistemi ile çevre koşullarına göre düşük düzeyde karar verebilme yetkisi.

Amerikan Robot Topluluğu ve İngiliz Robot Birliği tanımlamalarına göre, algılama sistemi olmayan yapıya robot denemez. İlk iki şartı yerine getiren yapıya programlanabilir manipülatör veya kısaca manipülatör, algılama sistemine de sahip olan manipülatöre de robot denir. Japonlar ise, robot tanımlamalarında ilk iki şartın yeterli olduğunu savunmaktadırlar [24].

Bilim adamı Asimov, 1939-40 yıllarında yazdığı romanında üç temel fikir olarak robotların sahip olması gereken fonksiyon ve sınırları şöyle tanımlamıştır:

1. Bir robot, insanlara zarar vermemeli, onlara zarar gelmesine seyirci kalmamalıdır.
2. Birinci kuralla çelişmediği sürece bir robot daima insanlardan aldığı emirlere uymalıdır.
3. Birinci ve ikinci kuralla çelişmediği sürece bir robot kendini, kendisine zarar verecek hareketlerden korumalıdır.

## *3.1 Robotların Sınıflandırılması*

Robotlar kendi aralarında çeşitli kıstaslara göre sınıflandırılmaktadırlar. Aşağıda en çok yapılan sınıflandırma örnekleri verilmiştir [24]:

### 3.1.1. Tahrik sistemlerine göre robotlar

Robotun önemli elemanlarından biri olan tahrik sistemi, robotun hareketini sağlar. Robotun kullanacağı alana ya da gerek duyduğu güce göre tahrik sistemleri üç çeşittir. Genellikle sanayide kullanılan bu sistemler:

1. Hidrolik sistem,
2. Elektrikli sistem,
3. Pnömatik sistem.

olarak sınıflandırılabilirler.

#### *3.1.1.1 Hidrolik sistem:*

Hidrolik tahrik sistemi, robota büyük hız ve güç verir. Bu sistem, mafsalların doğrusal ve dairesel hareket etmesini sağlayacak şekilde tasarlanır. Hidrolik sistemin dezavantajı robotun fazla yer işgal etmesidir. Ayrıca, hidrolik yağ sızarak ciddi kirlenme problemi oluşturmaktadır. Yüksek hız ve güç sağlandığından bu sistem birçok sanayi robotunda kullanılmaktadır. Sprey boyamadaki gibi elektrikli sistemlerin yangın çıkartma tehlikesi yüksek olan alanlarda hidrolik robotlar kullanılmaktadır.

#### *3.1.1.2 Elektrikli sistem:*

Hidrolik sistemler ile karşılaştırıldığında, elektrikli sistemler, daha az hız ve güç sağlarlar. Bu yüzden elektrikli sistemler daha küçük robotlarda kullanılır. Fakat bu sistemler daha doğru ve daha iyi tekrarlayabilme kabiliyetinde ve kullanımı daha temizdir. Sanayide en yaygın olarak bu tip robotlar kullanılır. Bu tip robotlar adım motorlular ve doğru akımlı servo motorlular olarak iki grupta sınıflandırılır. Adım motorlu robotların çoğu açık döngü tipindedir, fakat geri besleme döngüleri bu robotlarda ortaktır. Servo sistemli robotlar, sistem ile robot arasında sabit olan geri besleme döngülerine sahiptirler.

#### *3.1.1.3 Pnömatik sistem:*

Pnömatik tahrikli sistemler, genellikle daha küçük robotlarda kullanılır. Bu robotlar daha az serbestlik dereceli ve malzemeleri bir yerden alıp başka bir yere

nakletme işlemlerinde kullanılır. Bu işlemler genellikle basit ve kısa sürelidir. Pnömatik güç, doğrusal veya dairesel eklemler için kullanılır. Pnömatik robotlar, elektrikli veya hidrolik robotlardan daha ucuzdur. Çoğunlukla, pnömatik robotlar mekanik olarak her bir eksen için sabit noktalı işlemler yaparlar [24].

### 3.1.2. Koordinat sistemlerine göre robotlar

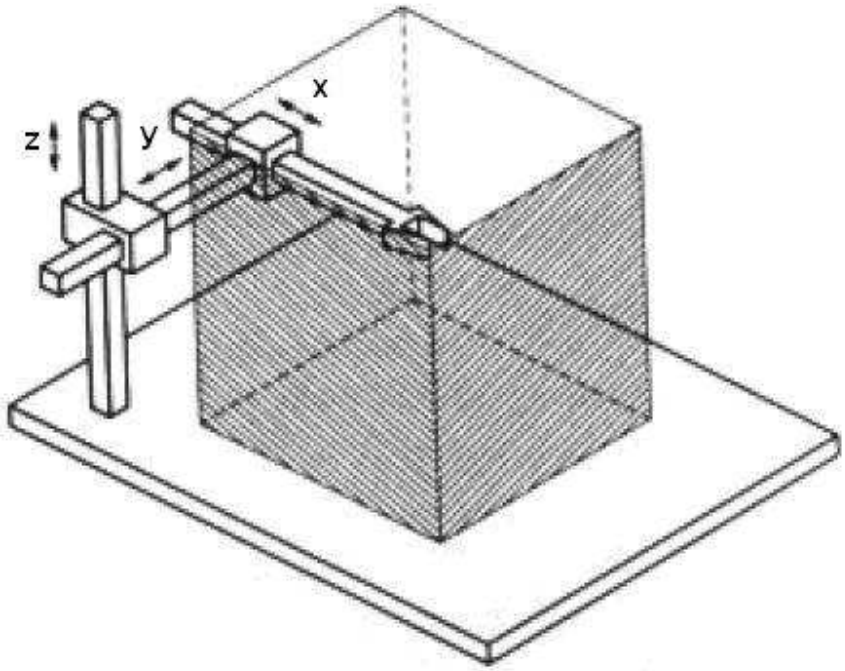
Koordinat sistemlerine göre robotlar dört kısımda incelenir:

1. Kartezyen koordinat sistemi,
2. Silindirik koordinat sistemi,
3. Küresel koordinat sistemi,
4. Döner koordinat sistemi.

#### *3.1.2.1 Kartezyen koordinat sistemi:*

Bu sistemde bütün robot uzuv hareketleri; birbirlerine karşı dik açılı şekilde olur (Şekil 3.1). Bu konfigürasyon en kısıtlı hareket serbestine sahip robot tasarım şeklidir. Bazı parçaların montajı için gerekli işlemler, kartezyen konfigürasyonlu robotlar tarafından yapılır. Hareketli kısımlar X, Y ve Z kartezyen koordinat sistemi eksenlerine paralel hareket ederler. Robot, üç boyutlu dikdörtgen prizması hacmi içinde hareket edebilir.

Bir kartezyen koordinat sisteminde, koordinat sistem merkezinin yeri, ilk iki bağlantının birleşme yerinin merkezidir. Merkezine doğru yapılan hareketler dışında, merkez hareket etmez, yani robotun merkezi sabittir. Robotun yerleştirildiği çalışma alanında eğer X yönündeki hattı bir kolona doğru çevrilirse, X hattı daima aynı kolona doğru yönelir, robotun programı yapılırken döndüğü yönde sorun yoktur. Bunlar, verilmiş bir robot donanımı için yer koordinatları olarak bilinir.




**Şekil 3.1 Kartezyen koordinat sistemi ve çalışma hacmi**

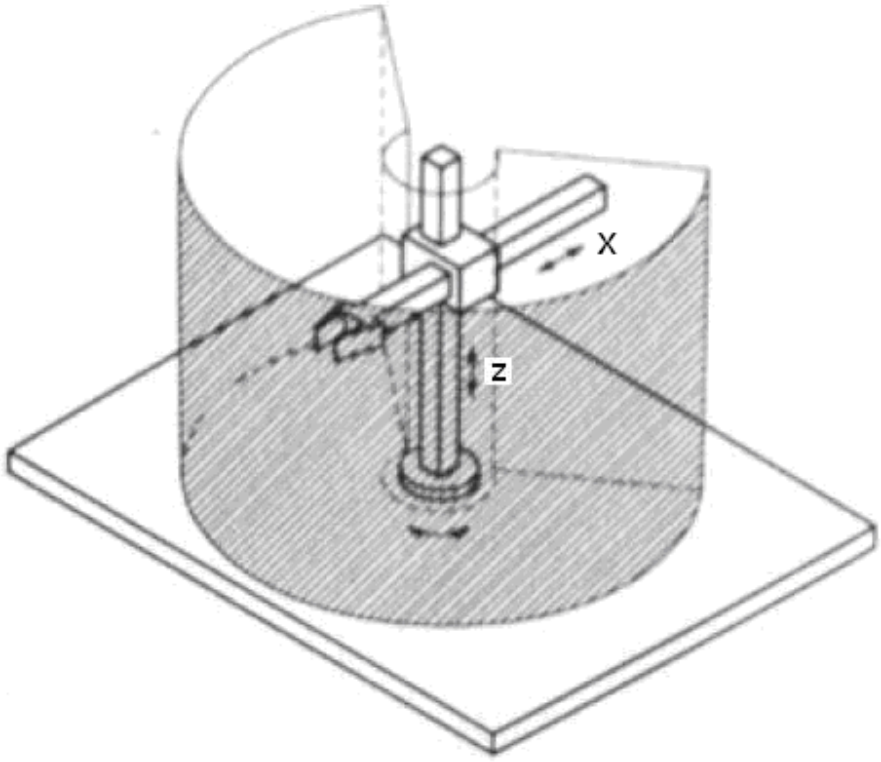
### *3.1.2.2 Silindirik koordinat sistemi:*

Bu tip robotlar temel bir yatak etrafında dönebilir ve diğer uzuvları taşıyan ana gövdeye sahip özelliktedir (Şekil 3.2). Hareket düşeyde ve ana gövde eksen kabul edildiğinde radyal olarak sağlanır. Dolayısıyla çalışma hacmi içerisinde robotun erişemeyeceği, ana gövdenin hacmi kadar bir bölge oluşur. Ayrıca genellikle, mekanik özelliklerden dolayı gövde tam olarak 360° dönemez.

Silindirik koordinatlarda tabana dik eksen etrafında dönme ve bu eksen üzerinde ötelenme yapılırken bu eksene dik bir eksende de başka bir öteleme hareketi yapılır. Dönme serbestliğindeki mekanik engellerden dolayı teorik olarak silindirik bir çalışma alanı oluşması beklenirken, bazı bölgelerde silindir yapısı tamamlanamaz. Zemine ulaşabilmenin arzu edildiği durumlarda robot kolu, zemine açılan bir yuvaya yerleştirilir. Ancak bu durumda da ulaşılabilecek maksimum yükseklik azalır. Radyal hareketten dolayı, silindirik koordinatlı robotlar montaj, kalıpçılık gibi alanlarda kullanılabilir. Bu tip robotlar da programlama açısından fazla karmaşık değildir. Ancak

kartezyen koordinatlı robotlarda olduğu gibi, kayar elemanların korozyon ve tozlanmadan korunması gerekir.

Silindirik robotlar genellikle, kendi eksenleri etrafında 300° dönmektedir. Geri kalan 60° lik alan ise robotun etrafında güvenli bir alan oluşturmak için kullanılır ve ölü bölge olarak adlandırılır.

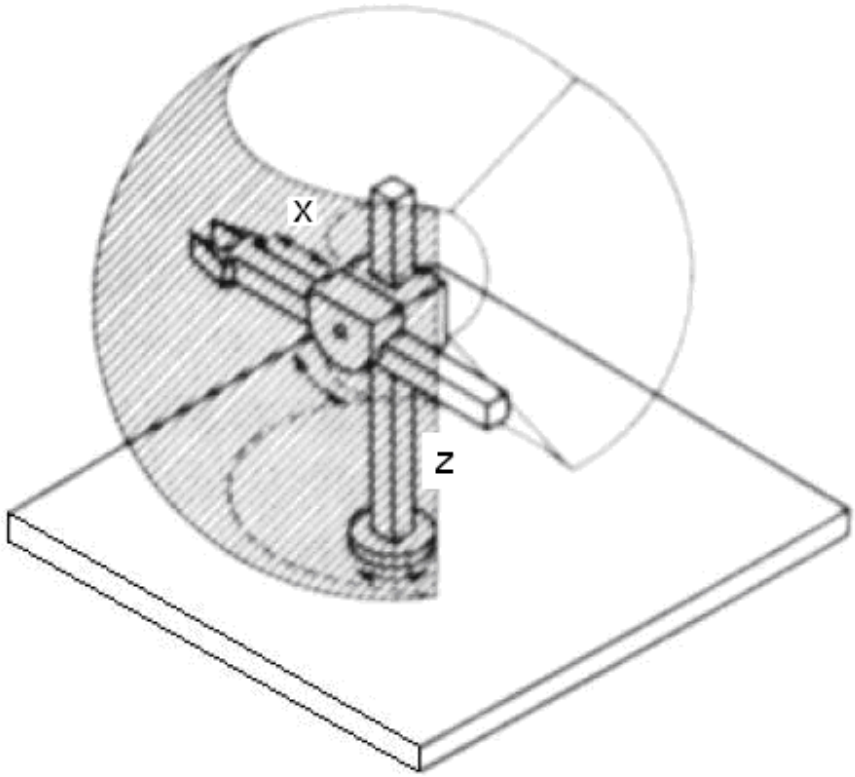



**Şekil 3.2 Silindirik koordinat sistemi ve çalışma alanı**

### *3.1.2.3 Küresel koordinat sistemi:*

Matematiksel olarak küresel koordinat sisteminin iki tane dairesel ve bir de doğrusal ekseni olmak üzere üç tane ekseni vardır (Şekil 3.3).

Robotikte küresel koordinat sistemi en eski koordinat sistemlerinden biridir. Oldukça çok işlevli, birçok uygulama alanına sahip özelliğinin yanında, yapım ve montaj açısından da oldukça kolaylık sağlamaktadır.



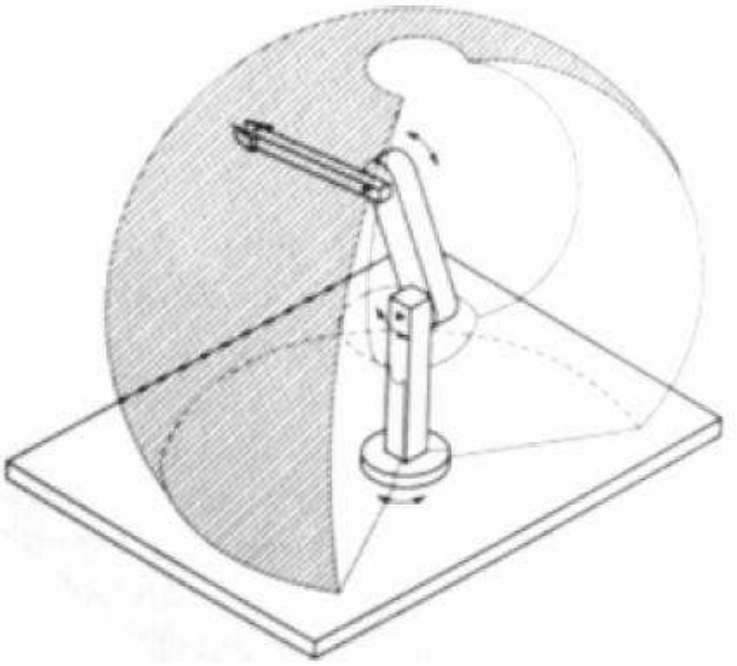

**Şekil 3.3 Küresel koordinat sistemi ve çalışma alanı**

Temelde yatay ve düşey dönme olarak iki hareketi mevcuttur. Üçüncü bir hareket ise doğrusal (uzama kolunun ileri geri hareketi) harekettir. Doğrusal hareket kartezyen koordinatlardan herhangi bir koordinatın hareketi gibi davranış gösterir.

Kutupsal koordinatlarda çalışan bir robotun çalışma hacmi iki kürenin ara hacminden oluşur. Koldaki uzuvlardan biri doğrusal hareket yaparken, bunu destekleyen diğer uzuvlardan biri tabana dik eksen etrafında, diğeri ise bu eksene dik ve tabana paralel eksen etrafında döner. Ölü bölgeler bu tip robotlarda da vardır. Öteleme hareketi yapan uzvun strokunun yetersizliğinden dolayı zemine ulaşmak mümkün olmaz.

### *3.1.2.4 Döner koordinat sistemi:*

Robot herhangi bir iş yaparken kolu, dairesel hareketli bağlamlarla oluşturuyorsa, bu tip robotlara döner koordinat sistemli robotlar denir.



**Şekil 3.4 Döner koordinat sistemi ve çalışma alanı**

Robot kolunun bağlantıları gövde üzerine, etrafında dönecek şekilde monte edilmiştir ve dayanak noktaları birbirine benzeyen iki ayrı bölümü taşır. Dönen parçalar yatay ve dikey monte edilebilir. 360° dönme sağlanamaz ancak bu kayıplar minimuma indirilebilir. Şekil 3.4' de döner koordinatlarda çalışma hacmi görülmektedir. Bu tip robotlarda robot kolun çalışması zor gözlenir. Çalışma hacmindeki noktalara farklı yörüngelerle ulaşılabilir. Buna göre sistem parametrelerinin en uygun olduğu yol seçilmelidir.

Döner koordinatlı robotlarda kontrol işlemi karmaşıktır, dolayısıyla kontrol donanımının da bu karmaşıklığı karşılayabilecek kapasitede olması gerekir. Ayrıca bu tip robotlarda mafsallarda sızdırmazlık kolayca sağlanabilmektedir [2], [9], [24].

### 3.1.3. Robot tiplerine göre sınıflandırma

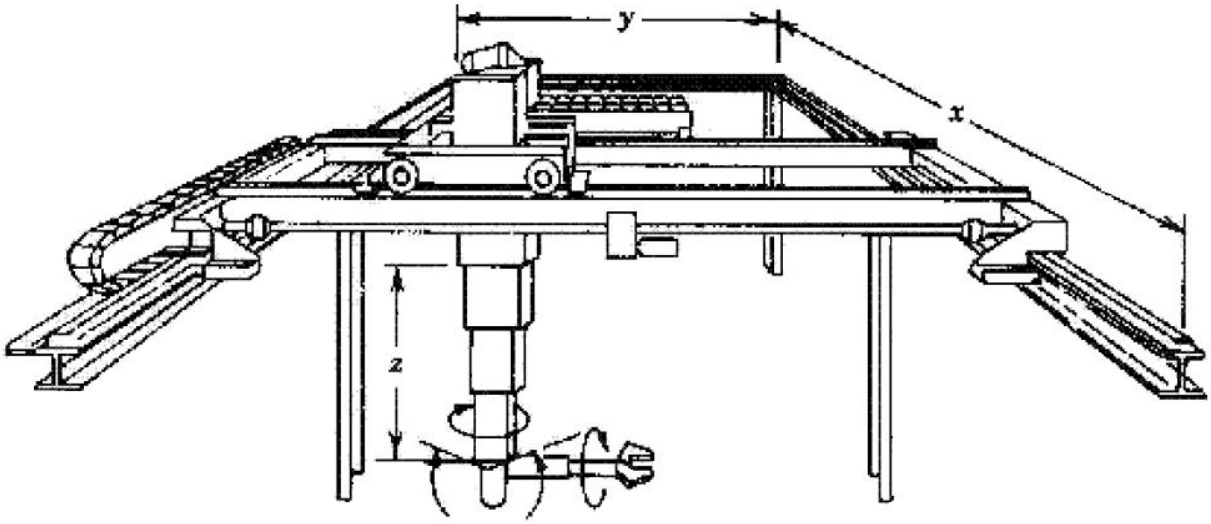
1. Kartezyen robotlar,
2. Mafsallı robotlar,

3. SCARA tipi robotlar.

### *3.1.3.1 Kartezyen robotlar:*

Kartezyen koordinat sisteminde bütün robot hareketleri birbirine 90’ lik açıyla hareket eder (Şekil 3.5).

Bu türün robotları, genellikle özel tatbiklerle sınırlandırılır. Devamlı bir yol alanında, robot, bir köprü ve bir ray sistemi aracılığıyla daha çok işlevlik kazanabilir. Tavana monte edilerek, birkaç fonksiyonla bir çok istasyona hizmet verilebilir. Robotun tavana asılı olmasıyla, zeminde daha fazla boş saha kazanılmış olur. Kartezyen robotlar, basitlikleri ve konstrüksiyonları sayesinde rijitliği yüksek ve hızlı bir yapıya sahiptir.



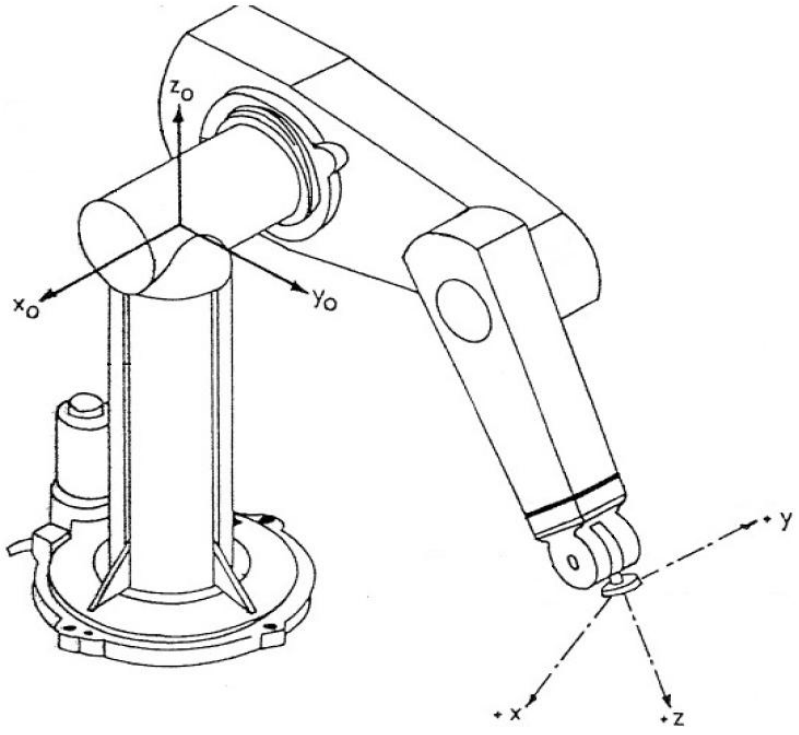

**Şekil 3.5 Kartezyen robot**

### *3.1.3.2 Mafsallı robotlar:*

Mafsallı robotların dizaynı insan kolundan esinlenerek yapılmıştır. Kol eklemli robotlar yeteneklerine göre, insan kolunun yerine getirebileceği görevleri üstlenmek

amacı ile yapılmışlardır. Kol eklemli robotlar, insan kollarında olan tüm esnekliğe ve hassasiyete tam olarak sahiptir ve değişik görevlerde insan kolunu taklit eder.

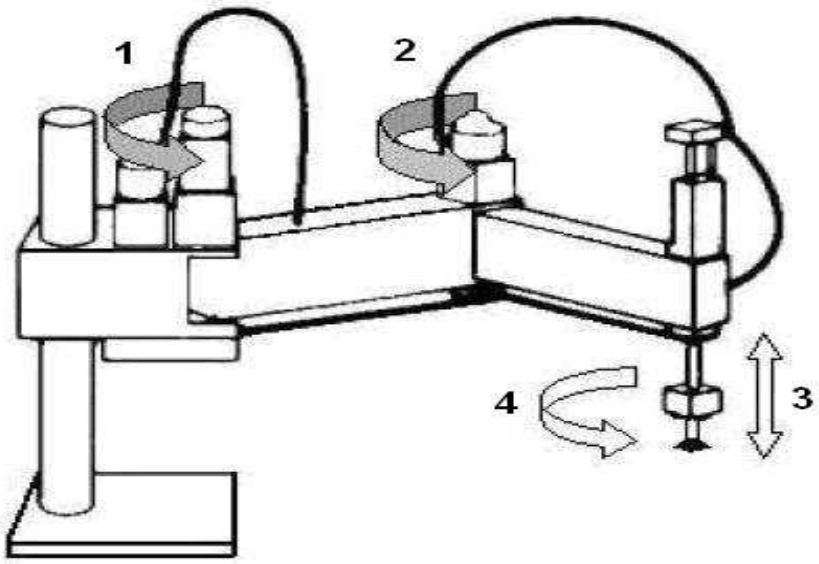
Kol eklemli robotlar altı eksende de rahatça hareket ederler. Bu altı eksenden üç tanesi kol hareketi için, diğer üç tanesi ise bilek hareketi içindir (Şekil 3.6). İnsan kolunun yapabileceği çok sayıda hareketi yapabilmektedirler. Bu özellikleri kullandıkları koordinat sisteminden (döner koordinat sisteminden) almaktadırlar. Bu koordinat sisteminin gereği olarak omuz, dirsek ve bilek bağlantıları vardır. Bu bağlantı şeklinin robota kazandırdığı en büyük avantaj, çalışma alanındaki her noktaya rahatça ulaşabilmesidir. Çalışma alanı ise; robot kolunun yatayda dik olarak durması sonucu elde edilir.




**Şekil 3.6 Mafsallı robot**

### *3.1.3.3 SCARA tipi robotlar:*

SCARA, “Selectively Compliant Articulated Robot Arm (Seçici Serbest Esnemeli Robot Kolu)” kelimelerinin baş harflerinden oluşmuştur. Bu robot 1970’ den sonra Japon Endüstriyel Konsorsiyumu ve bir grup araştırmacı tarafından Japonya’ da Yamanashi Üniversitesinde geliştirilmiştir. SCARA tipi robot, çok yüksek hıza ve iyi tekrarlama kabiliyetine sahip olan bir robot çeşididir.



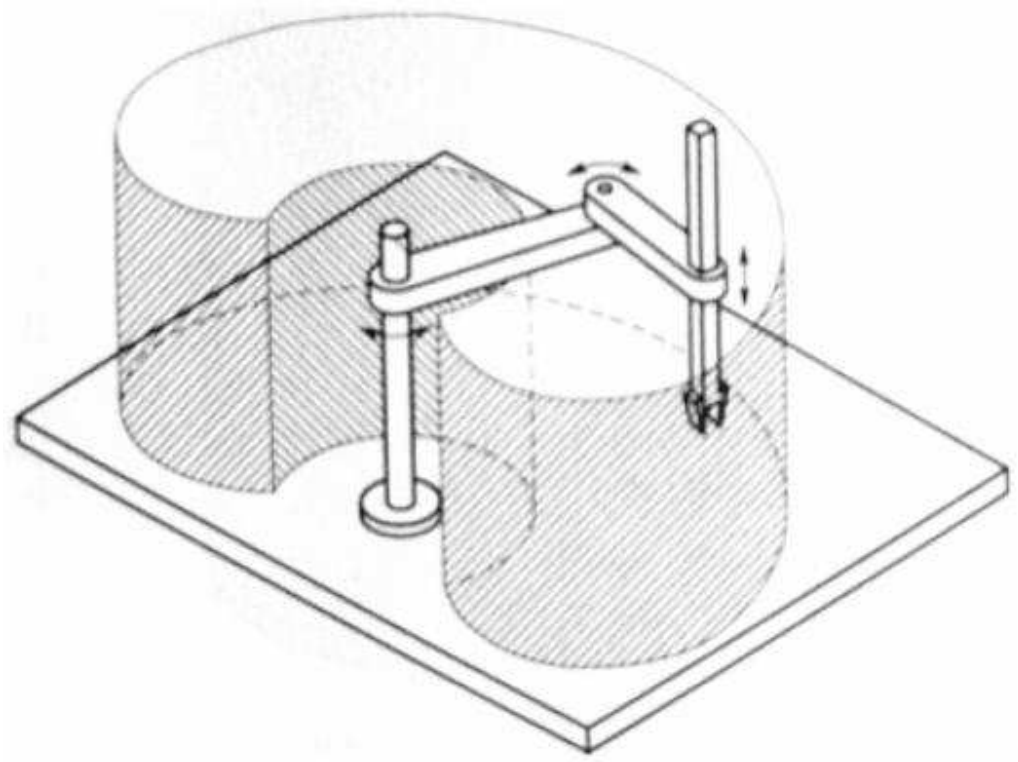

**Şekil 3.7 SCARA tipi robot**

Şekil 3.7’ de SCARA tipi bir robota ait şematik çizim verilmiştir. Bu robotta üç genel özellik bulunmaktadır:

1. Doğruluk,
2. Yüksek hız,
3. Kolay montaj.

Bu robot genellikle dikey eksen çevresinde dönen iki veya üç kol bölümünden meydana gelmiştir. Şekil 3.7’ de görülen 1 numaralı eksen, robota ana dönmeyi veren eksendir. Bu eksen en çok montaj robotlarında kullanılmaktadır. 2 numaralı eksende robot kolunun erişebileceği uzaklık değiştirilebilir. 3 numaralı eksen doğrusal dikey eksendir. Bu eksende sadece dikey hareket yapılabilmektedir. Bu özellik montaj robotlarında özellikle istenilmektedir. Dikey eksen hareketleri koordinat hareket

eksenleri içinde aşağıya doğru yapılan en çabuk ve düzgün hareketlerdir. 4 numaralı eksende ise dönen kol bileği hareket eder. Şekil 3.8’ de robotun çalışma alanına ait hacim verilmiştir [2], [9], [24].



**Şekil 3.8 SCARA robotun çalışma hacmi**

### 3.1.4. Kontrol sistemlerine göre robotlar

1. Sınırlı hareket eden robotlar,
2. Noktadan noktaya hareket eden robotlar,
3. Sürekli güzergâhlı robotlar,
4. Zeki robotlar.

#### *3.1.4.1 Sınırlı hareket eden robotlar:*

Eklemlerinin izafi pozisyonlarını göstermek için servo kontrol kullanmazlar. Bunun yerine, her eklemin hareketi boyunca yapacağı duruşlar kesiciler (switch) ile veya mekanik durdurucularla belirlenir. Pozisyonların ve duruş sınırlarının böyle belirlenmesi bir robot programlama olmayıp, mekanik bir ayarlama işlemidir. Bu tür kontrol sisteminde eklemler sadece limitleri içerisinde hareket edebilirler.

#### *3.1.4.2 Noktadan noktaya hareket eden robotlar:*

Arzu edilen bir dizi noktada hareket çevrimleri ve benzeri hareketler yapma yeteneğine sahiptirler. Önce her nokta robotun kontrol ünitesine kaydedilir. Hareket boyunca robot, bir noktadan ötekine istenilen sırada gidecek şekilde kontrol edilir. Burada robot, gidilen yolu takip etmez. Eğer programcı yolda küçük bir değişiklik yapmak isterse, robot programı yeniden değiştirilerek robota yüklenmelidir.

#### *3.1.4.3 Sürekli güzergâhlı robotlar:*

Robotun kontrol edildiği yol boyunca hareket çevrimleri yapma yeteneğine sahiptirler. Bu genellikle istenilen yolu tarif eden birbirine yakın noktaların takip edilmesiyle olur. Bu noktalar programcı yerine kontrol ünitesi tarafından sağlanır. Programcı sadece yolun başlangıcını ve bitimini verir. Kontrol ünitesi düz çizgiler oluşturacak şekilde noktalar belirler. Günümüzde bu işlemi yapan kontrol ünitesi olarak bilgisayarlar kullanılmaktadır.

#### *3.1.4.4 Zeki robotlar:*

Sadece programlanmış bir hareketi tekrar etmekle kalmazlar, ayrıca istenildiğinde zeki denebilecek bir şekilde çevresiyle etkileşimde bulunma yeteneğine de sahiptirler. Zeki robotlar, iş yerinde ortaya çıkan koşullara göre programlanmış çevrimini değiştirebilirler. Operasyonda elde edilen verilere göre mantıklı karar verebilirler [2], [9], [24].

# 4. ROBOTLARDA KİNEMATİK ANALİZ

Üç serbestlik dereceli bir robot kol; üç rijit uzvun ucu açık bir mekanizma oluşturacak şekilde mafsallarla birleştirilerek tahrik elemanları tarafından hareket ettirilmesi şeklinde modellenebilir.

Bir robot kolu uzvu, referans koordinat takımına göre dönme ve öteleme şeklinde iki temel hareket yapabileceğinden, uzvun hareketini tanımlamak için her bir uzvun mafsal ekseni üzerinde bir koordinat takımının bulunduğu düşünülecektir. Robotik uygulamalarda genellikle uç noktanın, üç boyutlu koordinat sisteminde sabit yer koordinatlarına göre tanımımın yapılması istenir.

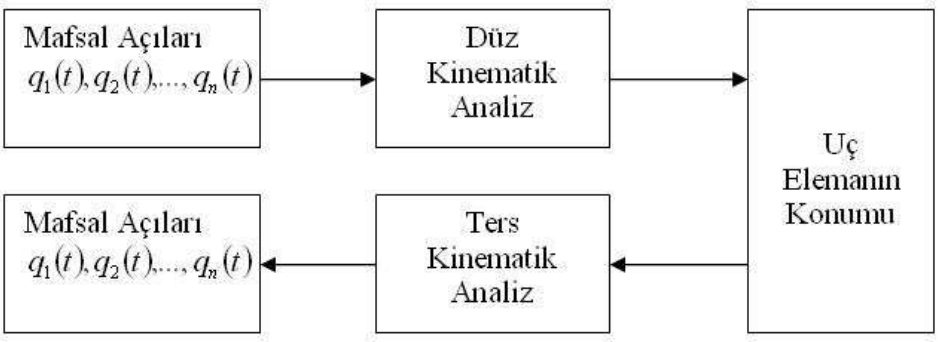
Robot kolun bir ucu zemindeki destek elemanına bağlıdır ve diğer ucunda robot koluna yaptırılacak işe uygun olarak bir takım bağlanmıştır. Mafsallardan verilen tahrikler sonucu manipülatörün uç noktası istenen konuma ve yönlenmeye götürülür.

Kinematik analiz çözümü için iki farklı yöntem bulunmaktadır:

1. Verilen bir manipülatör için mafsallara ait açılar vektörü $q(t) = (q_1(t), q_2(t), ....., q_n(t))^T$ ve geometrik kol parametreleri verilir ve uç noktanın sabit eksen takımlarına göre konumu ve yönlenmesi nedir? sorusunun cevabı aranır. Bu problem düz kinematik problem olarak adlandırılır.

2. Uç noktanın istenen konumu ve yönlenmesi sabit eksen takımına göre kol parametreleriyle birlikte verilerek manipülatör bu noktaya ulaşabilir mi, ulaşabilirse kaç çeşit robot kol konfigürasyonu bu şartları sağlar? sorularının cevabı aranır. Bu probleme ise ters kinematik problem adı verilmektedir.

Bir robot kolunda bağımsız değişkenler mafsal değişkenleri olduğundan ve yapılacak iş referans yer koordinatlarına göre tanımlandığından ters kinematik analiz daha sık kullanılır. Her iki durumun blok diyagramı Şekil 4.1' de görülmektedir.




**Şekil 4.1 Düz ve ters kinematik problemleri**

Robot kolun uzuvları referans koordinat takımına göre, ya dönme ya da öteleme hareketi yapar. Bu nedenle uç elemanın üç boyutlu uzaydaki toplam yer değiştirmesi uzuvların açısal dönmeleri ve doğrusal ötelenmeleri sonucu oluşur. Denavit ve Hartenberg, 1955' de bir metot geliştirerek, referans eksen takımına göre uzuvların uzaysal geometrisinin, matris cebri ile gösterilmesini sağlamışlardır. Bu metot 4x4 homojen dönüşüm matrislerini kullanarak birbirine komşu iki mekanik uzva ait uzaysal ilişkileri tanımlar.

Ters kinematik problem, genelde birkaç metotla çözülebilir. En çok kullanılan metotlar matris cebri metodu, iterasyon ve geometrik yaklaşım metotlarıdır [2], [9], [29].

## *4.1. Düz Kinematik Analiz*

Robot kolunun geometrik uzuv parametreleri (D-H Parametreleri) ve zamanla değişen eklem değerleri $(\theta_i, d_i)$ kullanılarak, uç elemanın konum ve yönlenmesinin hesaplanmasıdır [2].

### 4.1.1. Denavit – Hartenberg yöntemi

Bir mafsal ekseni (i mafsalı için), iki uzvun birleşim noktasına yerleştirilir. Eklemlere koordinat sistemi yerleştirilirken sırayla aşağıdaki işlemler gerçekleştirilir:

1. Eklem eksenlerinin dönme ve öteleme yönleri belirlenir ve bu eksene paralel bir doğru çizilir.
2. Dönel eksenler için dönme yönü ve prizmatik eklemler için öteleme yönü Z olarak belirlenir.
3. Z eksenine dik ve kol boyunca olan eksen X ekseni olarak belirlenir.
4. Sağ el kuralına göre Y ekseni belirlenir.
5. Sıfır ve birinci eklemler üst üste kabul edilebilir.
6. Bir seri robotun eklemlerine koordinat sistemleri yerleştirilirken, birinci eksenin dönme yönü Z ekseni olarak belirlenir. X ekseni döndürüldüğünde komşu iki Z ekseni üst üste çakışacak şekilde bir X ekseni yerleştirilir [2].

Bir uzva eksen takımının yerleştirilmesi şöyle yapılır ( Şekil 4.2 ): {i} eksen takımının z ekseni yani $Z_i$, eklem ekseni ile çakışacak şekildedir. Eksen takımının orijini ise $a_i$ ile eklem ekseninin kesişim noktasında olur. $X_i$ ise $a_i$ doğrultusunda ve $i$' den $i+1$' e doğru olacak şekilde yerleştirilir. $Y_i$ ise sağ el kuralı ile bulunur. $a_i = 0$ iken $X_i$' nin yön seçiminde birden fazla seçenek ile karşılaşmak mümkündür [12].

Eksen takımları açıklandığı gibi yerleştirilirse, uzuv parametreleri şöyle tanımlanabilir [2], [12]:

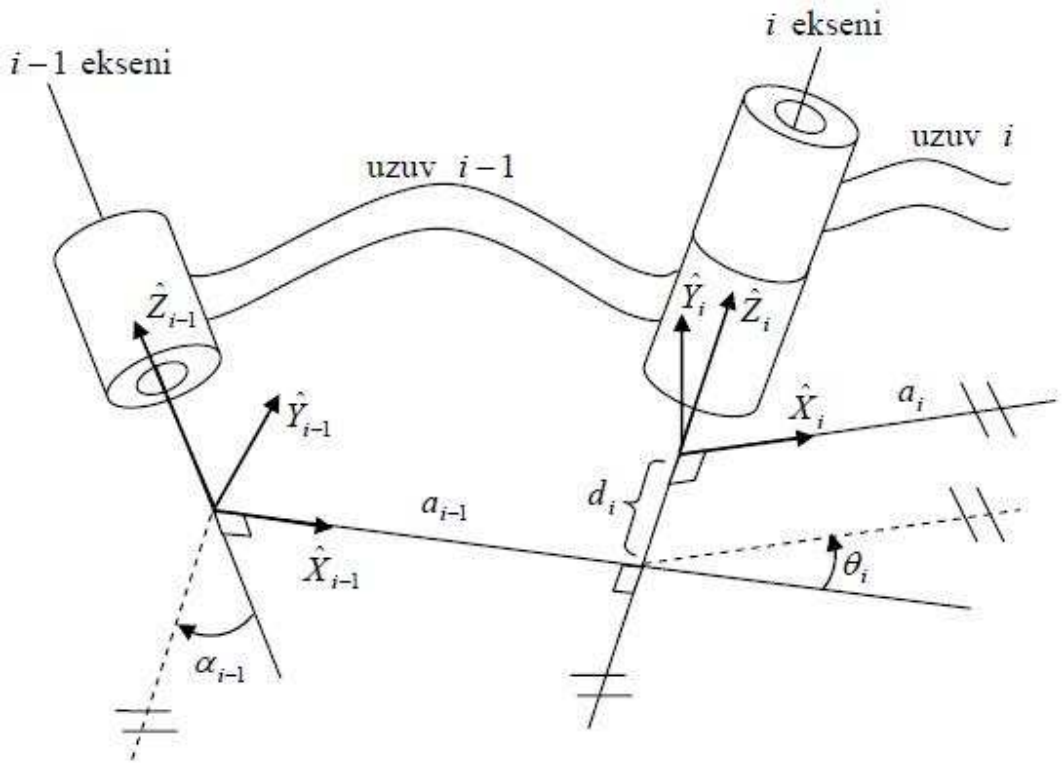
$a_i$ : $Z_i$' den $Z_{i+1}$' e, $X_i$ boyunca ölçülen mesafe.

$\alpha_i$ : $Z_i$' ile $Z_{i+1}$ arasında, $X_i$ etrafında ölçülen açı.

$d_i$ : $X_{i-1}$' den $X_i$' ye, $Z_i$ boyunca ölçülen mesafe.

$\theta_i$ : $X_{i-1}$' den $X_i$ eksenine, , $Z_i$ etrafında ölçülen açı.

$a_i$ genelde sıfırdan büyüktür, diğer üç parametrenin işareti ise uzvun doğrusal ve açısal konumlarına göre değişebilir.




**Şekil 4.2 Eksen takımlarının uzuvlara yerleştirilmesi**

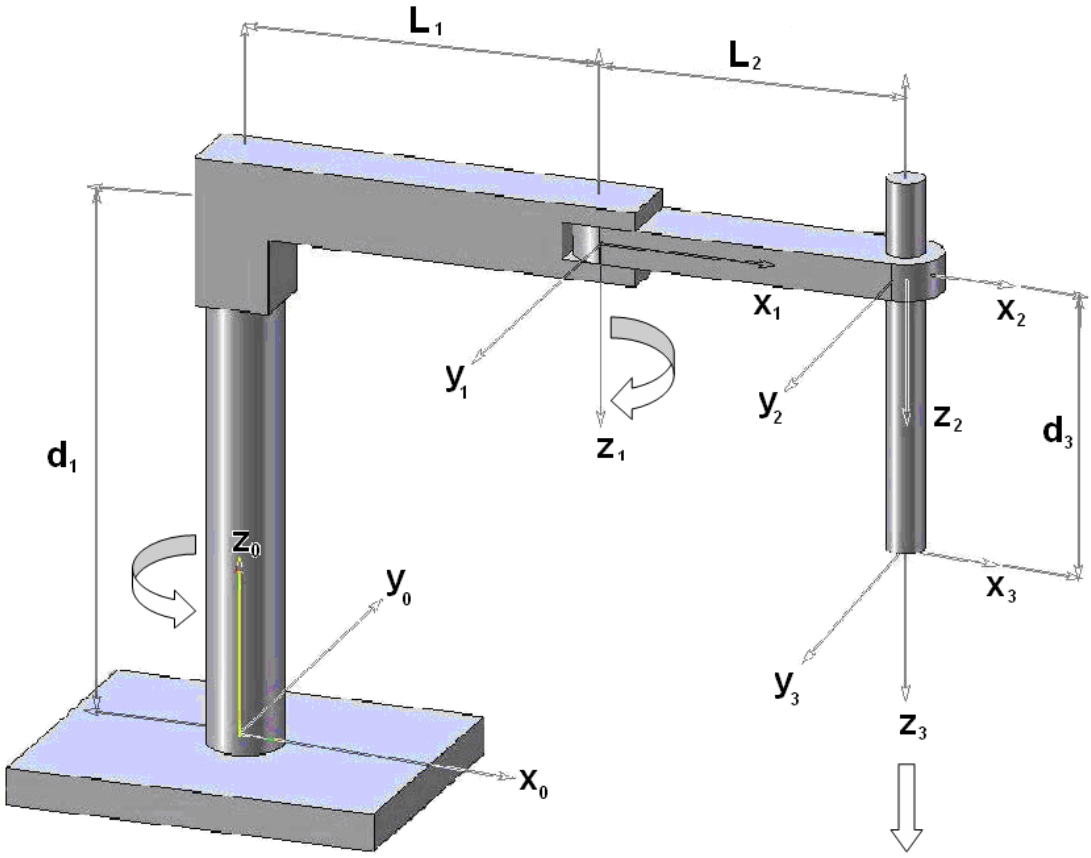
Kinematik açıdan uzuvların önemi, mafsalları arasında sabit bir konfigürasyon oluşturmalarıdır. Bu konfigürasyon $a_i$ ve $\alpha_i$ gibi iki parametre ile tanımlanabilir. $a_i$ parametresi, mafsal eksenleri arasındaki ortak normalleri boyunca ölçülen en kısa

mesafedir (i ve i+1 mafsalları için $Z_i$ ve $Z_{i-1}$ eksenleri), $\alpha_i$ ise mafsal eksenleri arasındaki $a_i$' ye dik bir düzlem üzerinde ölçülen açıdır. Bu durumda $a_i$ ve $\alpha_i$, i uzvunun boyu ve dönme açısı olarak bilinir ve i uzvunun yapısını belirler [2], [7], [9].

Uzuvların birbirleriyle bağlantısını belirlemek için iki parametre tanımına daha ihtiyaç vardır, uzuv açıklığı $(d_i)$ ve eklem açısı $(\theta_i)$. Uzuv açıklığı, $a_{i-1}$'in ekseni kestiği noktadan, $a_i$'nin ekseni kestiği noktaya olan uzaklıktır. Eklem açısı ise $a_{i-1}$' in uzantısı ile $a_i$ arasındaki açıdır. Böylece bir robotun kinematik tanımlaması, her uzuv için dört parametrenin verilmesiyle yapılabilir. Bunlardan ikisi uzvun kendisini, diğer ikisi de uzuvların birbiriyle olan ilişkilerini tanımlar. Eğer dönen eklem söz konusu ise $\theta_i$ eklem değişkeni, diğer üç parametre ise sabit parametreler olur. Eğer eklem prizmatik ise $d_i$ eklem değişkeni, diğer üç parametre ise sabit parametreler olur.

Bu parametreleri kullanarak mekanizmaları tanımlamak için, Denavit – Hartenberg yöntemi kullanılmaktadır. Bu yöntemde her uzva bir eksen takımı yerleştirilir. Böylece bu eksen takımları sayesinde uzuvların birbirine göre durumları tanımlanabilir. Bu yöntemde eksen takımları yerleştirildikleri uzva göre isimlendirilirler. $\{i\}$ eksen takımı $i$ uzvuna yerleştirilmiş demektir. Numaralandırma sıfırdan başlayarak yapılır. Sıfır numaralı eksen takımı, robot gövdesinin hareket etmeyen bir yerine yerleştirilir [2], [12].

Kontrolü düşünülen robot üç serbestlik derecesine sahiptir. Sistem değişkenleri $(\theta_1, \theta_2, d_3)$' dür. Robotun uzuv koordinat sistemi ve mafsal koordinatlarının yerleşimleri Şekil 4.3' de gösterilmiştir.




**Şekil 4.3 Uzuv koordinat sistemi**

**Tablo 4.1 Robotun bu konfigürasyonu için D – H çizelgesi**

| Eksen | $\theta$ | d | a (L) | α | Başlangıç |
|---|---|---|---|---|---|
| 1 | $\theta_1$ | $d_1$ | $L_1$ | π | 0 |
| 2 | $\theta_2$ | 0 | $L_2$ | 0 | 0 |
| 3 | 0 | $d_3$ | 0 | 0 | 90 |

Kontrolü düşünülen SCARA tipi robot için kinematik parametrelerin değerleri

$$d = [90 \quad 0 \quad d_3]^T \text{ mm} \tag{4.1}$$

$$a = [100 \quad 100 \quad 0]^T \text{ mm} \tag{4.2}$$

şeklindedir.

Elde edilen D – H parametrelerine göre robotun uç elemanının tabana göre konum ve oryantasyonunu gösteren transformasyon matrisi;

$$T_{taban}^{takim} = T_0^1 \cdot T_1^2 \cdot T_2^3 \tag{4.3}$$

formülündeki gibidir. Tablo 4.1' deki D – H parametrelerini kullanarak robota ait dönüşüm matrisleri aşağıdaki gibi elde edilir:

$$ {}_1^0T = \begin{bmatrix} \cos\theta_1 & -\sin\theta_1 & 0 & 0 \\ \sin\theta_1 & \cos\theta_1 & 0 & 0 \\ 0 & 0 & 1 & d_1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad {}_2^1T = \begin{bmatrix} \cos\theta_2 & -\sin\theta_2 & 0 & L_1 \\ \sin\theta_2 & \cos\theta_2 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$ {}_3^2T = \begin{bmatrix} 1 & 0 & 0 & L_2 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & -1 & -d_3 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.4}$$

Buna göre verilen matrisler eşitlik (4.3)' de yerine koyularak robot transformasyon matrisi elde edilebilir:

$$T_{taban}^{takıa} = \begin{bmatrix} \cos\theta_1 & -\sin\theta_1 & 0 & 0 \\ \sin\theta_1 & \cos\theta_1 & 0 & 0 \\ 0 & 0 & 1 & d_1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} \cos\theta_2 & -\sin\theta_2 & 0 & L_1 \\ \sin\theta_2 & \cos\theta_2 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & 0 & L_2 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & -1 & -d_3 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Matrisler çarpılarak sadeleştirildikten sonra sonuç matrisi;

$$\begin{bmatrix} C_{12} & S_{12} & 0 & L_1C_1 + L_2C_{12} \\ S_{12} & -C_{12} & 0 & L_1S_1 + L_2S_{12} \\ 0 & 0 & -1 & d_1 - d_3 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.5}$$

olarak bulunur [2] .

## *4.2. Ters Kinematik Analiz*

Ters kinematik analiz, kartezyen uzayda ana çerçeveye göre verilen uç elemanın konum ve yönelim verileri yardımıyla eklem değişkenlerinin bulunması şeklinde ifade edilebilir. Genelde robotun izlemesi istenen yörünge bilinir ve bu yörüngeyi sağlayacak mafsal değişkenlerinin bulunması gerekir. Bu da ters kinematik analiz ile sağlanır.

Kinematik denklemlerin içerdiği nonlineerlikler yüzünden çözüm yöntemleri, düz kinematik çözümlemeye göre daha karmaşıktır ve lineer denklem çözümünde olduğu gibi genel bir çözüm yöntemi yoktur. Bunun yanı sıra çözümün var olup olmadığı ve birden fazla çözümün varlığı gibi problemler de söz konusudur. Çözümün var olabilmesi için istenen robot eli konumunun, robotun çalışma hacmi içinde olması gerekmektedir. Aksi takdirde çözüm yoktur. Prizmatik eklemler çözüm sayısının azalmasına, dönel eklemlerse artmasına neden olmaktadır. Buna karşın, dönel eklemlerden oluşan robotlarda fiziksel çözüm sayısının fazla olması, üç boyutlu uzayda bir noktaya pek çok farklı şekilde ulaşmaya imkân sağlar.

Kullanılan çözüm yöntemlerini kapalı form çözümler ve sayısal çözümler olarak mümkündür. Sayısal çözümler iterasyonların uzun sürmesi yüzünden pek tercih edilmez. Kapalı form çözümler genelde analitik ifadelerden ya da dördüncü mertebeye kadar olan polinomlardan oluşur. Kapalı form çözümleri de kendi içinde cebirsel ve geometrik olmak üzere ikiye ayrılabilir ancak geometrik ifadelerde cebirsel ifadeler de kullanıldığından bu ayrım çok net yapılamaz.

Takımın konum ve oryantasyonunu belirleyen takım konfigürasyon vektörü kullanılarak gerekli mafsal değişkenleri bulunabilir [2], [9].

### 4.2.1. Mafsal değişkenlerinin bulunması

a. Ana Kol Mafsal Değişkeni $\theta_1$' in Bulunması

$${}^0_3T = {}^0_1T \cdot {}^1_2T \cdot {}^2_3T \qquad \left[{}^0_1T\right]^{-1} \cdot {}^0_3T = {}^1_2T \cdot {}^2_3T$$

$$\begin{bmatrix} . & . & . & \cos\theta_1 \cdot P_x + \sin\theta_1 \cdot P_y \\ . & . & . & -\sin\theta_1 \cdot P_x + \cos\theta_1 \cdot P_y \\ . & . & . & P_z - d_1 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} . & . & . & L_2 \cdot \cos\theta_2 + L_1 \\ . & . & . & L_2 \cdot \sin\theta_2 \\ . & . & . & -d_3 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.6}$$

$$\begin{aligned} &\cos\theta_1 \cdot P_x + \sin\theta_1 \cdot P_y = L_2 \cdot \cos\theta_2 + L_1 \\ &-\sin\theta_1 \cdot P_x + \cos\theta_1 \cdot P_y = L_2 \cdot \sin\theta_2 \end{aligned} \tag{4.7}$$

$$\begin{aligned} &\cos\theta_1 \cdot P_x + \sin\theta_1 \cdot P_y - L_1 = L_2 \cdot \cos\theta_2 \\ &-\sin\theta_1 \cdot P_x + \cos\theta_1 \cdot P_y = L_2 \cdot \sin\theta_2 \end{aligned}$$

Her iki tarafın kareleri alınarak toplanırsa;

$$\begin{aligned} &\cos^2\theta_1 P_x^2 + \sin^2\theta_1 P_y^2 + 2\cos\theta_1 \sin\theta_1 P_x P_y - 2\cos\theta_1 P_x L_1 - 2\sin\theta_1 P_y L_1 + L_1^2 = L_2^2 \cos^2\theta_2 \\ +\ &\sin^2\theta_1 P_x^2 + \cos^2\theta_1 P_y^2 - 2\cos\theta_1 \sin\theta_1 P_x P_y = L_2^2 \sin^2\theta_2 \end{aligned}$$

$$\begin{aligned} &P_x^2\left(\sin^2\theta_1 + \cos^2\theta_1\right) + P_y^2\left(\sin^2\theta_1 + \cos^2\theta_1\right) - 2\cos\theta_1 P_x L_1 - 2\sin\theta_1 P_y L_1 + L_1^2 \\ &= L_2^2\left(\sin^2\theta_2 + \cos^2\theta_2\right) \end{aligned} \tag{4.8}$$

$$2\cos\theta_1 P_x L_1 + 2\sin\theta_1 P_y L_1 = P_x^2 + P_y^2 + L_1^2 - L_2^2$$

$P_x^2 + P_y^2 + L_1^2 - L_2^2 = k$ yazılırsa;

$$\theta_1 = A\tan 2\left(P_y, P_x\right) \pm A\tan 2\left(\sqrt{4L_1^2\left(P_x^2 + P_y^2\right) - k^2}, k\right) \tag{4.9}$$

olarak elde edilir. Ters kinematik çözüm gerçekleştirilirken bazı trigonometrik eşitliklerden yararlanılır. EK-3' de verilen eşitliklerden 6. ifade çözümde kullanılmıştır. [2].

b. Ön Kol Mafsal Değişkeni $\theta_2$ ' nin Bulunması

Denklem (4.7)' de verilen eşitliklerde $\sin\theta_2$ ve $\cos\theta_2$ ifadeleri yalnız bırakılırsa;

$$\begin{aligned}
&\cos\theta_1 \cdot P_x + \sin\theta_1 \cdot P_y = L_2 \cdot \cos\theta_2 + L_1 \Rightarrow \\
&\cos\theta_2 = \frac{\cos\theta_1 \cdot P_x + \sin\theta_1 \cdot P_y - L_1}{L_2} \\
&-\sin\theta_1 \cdot P_x + \cos\theta_1 \cdot P_y = L_2 \cdot \sin\theta_2 \Rightarrow \\
&\sin\theta_2 = \frac{-\sin\theta_1 \cdot P_x + \cos\theta_1 \cdot P_y}{L_2}
\end{aligned} \tag{4.10}$$

$$\theta_2 = A\tan 2\left(\frac{-\sin\theta_1 \cdot P_x + \cos\theta_1 \cdot P_y}{L_2}; \frac{\cos\theta_1 \cdot P_x + \sin\theta_1 \cdot P_y - L_1}{L_2}\right) \tag{4.11}$$

olarak bulunur [2].

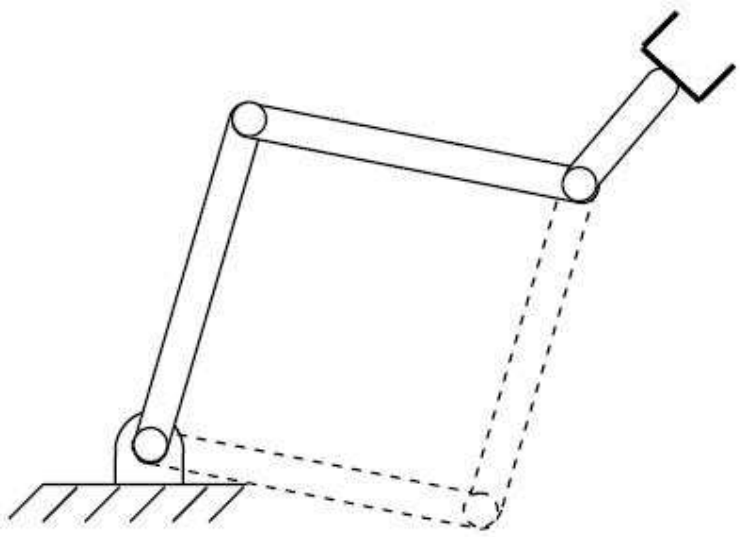
c. Uç Eleman Lineer Hareket Miktarı $d_3$ 'ün Bulunması

$$\begin{aligned}
&P_z - d_1 = -d_3 \\
&d_3 = d_1 - P_z
\end{aligned} \tag{4.12}$$

ifadesi elde edilir [2].

Takım uç noktası için ters kinematik analizle bulunan mafsal değişkenlerinin $(d_i, \theta_i)$ bir noktada birden fazla çözümü ortaya çıkabilmektedir. Bu problemi aşmak için öncelikle mafsal limitleri belirlenmelidir. Ayrıca uzuvların yapabileceği hareketler için belli kurallar konulmalıdır. Şekil 4.4' de görüldüğü gibi çalışma alanı içerisinde

istenen takım koordinatında iki çözüm bulunmaktadır. Mafsal açılarının sağ kol veya sol kol olacak şekilde seçilmesiyle çözüm kümesi daraltılmış olacaktır [9], [29].



**Şekil 4.4 Robotun sağ kol veya sol kol olarak tanımlanması**

### 4.2.2. SCARA robot için çalışma uzayının tanımlanması

Çalışma uzayının tayininde öncelikli olarak robot konstrüksiyon kısıtlarının bilinmesi gerekmektedir. SCARA tipi manipülatörlerde ana kol dönmesi $(\theta_1)$ genelde kısıtlayıcı bir etken içermez. Bu mafsal değişkeni için $-\pi \le \theta_1 \le \pi$ aralığında olacak şekilde bir kısıtlama yapılabilir. Ancak dirsek değişkeni $(\theta_2)$, robotun konstrüksiyonuna bağlı olarak sınırlanmalıdır. $\beta$ sınır açısı kabul edilirse, $(\theta_2)$ mafsal değişkeninin sınırları $-\pi + \beta \le \theta_1 \le \pi - \beta$ şeklinde olacaktır. Mafsal değişkenlerinin sınırlarının genel gösterimi (4.10) denkleminde görüldüğü gibidir [2], [29].

$$\begin{bmatrix} -2\pi \\ -\pi+\beta \\ h \end{bmatrix} \le \begin{bmatrix} \theta_1 \\ \theta_2 \\ d_3 \end{bmatrix} \le \begin{bmatrix} 2\pi \\ \pi-\beta \\ H \end{bmatrix} \tag{4.10}$$

Ulaşılabilir çalışma uzayı (reachable workspace) ve dexterious çalışma uzayı, robot manipülatörlerinin çalışma uzaylarını belirleyen çok önemli iki özelliktir. Ulaşılabilir çalışma uzayı, bir robot manipülatörünün uç işlevcisini rastgele hareket ettirip yönlendirdiği, robotların serbestlik derecelerinin azalmasına neden olan tekil noktaların bulunmadığı bölgeye denir. Dexterious çalışma uzayı ise, uç işlevcisinin yönelme ve öteleme hareketlerini en büyük kapasitede gerçekleştirdiği bölgedir. Dolayısı ile dexterious çalışma uzayı, ulaşılabilir çalışma uzayının bir alt kümesidir. İyi tasarlanmış bir robot, en kısa bağ uzunluğuna sahip olmasına rağmen en büyük hacimli bir çalışma uzayını tarayan ve bu hacim içerisinde en iyi hareket kabiliyetinin gerçekleştirildiği en büyük dexterious çalışma uzayına sahip olan robottur [2].

# 5. YÖRÜNGE PLANLAMASI

Yörünge planlaması robotun uç elemanının bulunduğu konumdan arzu edilen konuma, titreşimden uzak, çalışma uzayındaki herhangi bir cisme çarpmadan, eyleyicilerin sınırlarını zorlamadan kontrollü ve yumuşak bir şekilde hareket ederek ulaşması için yapılmaktadır.

Robot manipülatörler için kartezyen ve eklem uzayı olmak üzere iki farklı yaklaşımla yörünge planlaması yapılmaktadır [2].

Bu çalışmada eklem uzayında yörünge kontrolü yöntemi kullanılmıştır.

## *5.1. Eklem Uzayında Yörünge Kontrolü*

Eklem uzayında yörünge planlaması yapılırken üç veya daha yüksek dereceli polinomlar kullanılır. Örneğin; uç eleman başlangıç noktasından arzu edilen noktaya belli bir zaman aralığında gitsin. İlk olarak uç elemanın başlangıç ve hedef noktalarının konumu ve yönelimi ters kinematik analiz ile eklem açıları cinsinden hesaplanır. Uç elemanın $t_0$ anındaki başlangıç konumu $\theta(0)=\theta_0$ ve $t_f$ anındaki hedef konumu $\theta(t_f)=\theta_f$ olsun. Bu durumda $t_0$ ve $t_f$ arası üçüncü dereceden bir polinom yardımıyla n tane noktaya bölünür. Bu iki koşula ek olarak başlangıç ve bitiş hızları $\dot{\theta}(0)=0$ ve $\dot{\theta}(t_f)=0$ eklenir [2].

$\theta(0)=\theta_0$ (Uç işlevcisinin $t_0$ anındaki başlangıç konumu) (5.1)

$\theta(t_f)=\theta_f$ (Uç işlevcisinin $t_f$ anındaki hedef konumu) (5.2)

$\dot{\theta}(0)=0$ (Uç işlevcisinin $t_0$ anındaki başlangıç hızı) (5.3)

$\dot{\theta}(t_f)=0$ (Uç işlevcisinin $t_f$ anındaki bitiş hızı) (5.4)

Yukarıdaki dört koşul, üçüncü dereceden bir polinomun katsayılarını bulmak için yeterlidir. Bu koşullar , zamana bağlı kübik bir yörünge oluşturan üçüncü dereceden polinom olarak;

$$\theta(t) = s_0 + s_1 t + s_2 t^2 + s_3 t^3 \tag{5.5}$$

şeklinde ifade edilir. Bu yörüngedeki eklem hızları ve ivmeleri, (5.2) denkleminin birinci ve ikinci türevleri alınarak bulunur:

$$\dot{\theta}(t) = s_1 + 2s_2 t + 3s_3 t^2 \tag{5.6}$$

$$\ddot{\theta}(t) = 2s_2 + 6s_3 t \tag{5.7}$$

(5.1) ve (5.2) denklemleri, (5.5) denkleminde yerine yazılarak aşağıdaki ifadeler elde edilir:

$$\begin{aligned} &\theta(0) = s_0 + s_1 \cdot 0 + s_2 \cdot 0^2 + s_3 \cdot 0^3 \\ &s_0 = \theta(0) = \theta_0 \end{aligned} \tag{5.8}$$

$$\begin{aligned} &\theta(t_f) = \theta_f \\ &\theta_f = s_0 + s_1 t_f + s_2 {t_f}^2 + s_3 {t_f}^3 \end{aligned} \tag{5.9}$$

(5.3) ve (5.4) denklemleri, (5.6) denkleminde yerine yazılırsa aşağıdaki ifadeler bulunur.

$$\begin{aligned} &\dot{\theta}(0) = s_1 + 2s_2 \cdot 0 + 3s_3 \cdot 0^2 = 0 \\ &s_1 = 0 \end{aligned} \tag{5.10}$$

$$\dot{\theta}(t_f) = s_1 + 2s_2 t_f + 3s_3 {t_f}^2 = 0 \tag{5.11}$$

Polinomun katsayıları yukarıda elde edilen denklemlerden faydalanarak bulunur. $s_0$ ve $s_1$ katsayıları, sırasıyla (5.8) ve (5.10) denklemlerinden bulunmuştur. $s_2$ katsayısını bulmak için, (5.11) denkleminde, $\dot{\theta}(t_f)=0$ ve $s_1=0$ değerleri yerine yazılarak;

$$\dot{\theta}(t_f)=s_1+2s_2t_f+3s_3{t_f}^2=0$$
$$0+2s_2t_f+3s_3{t_f}^2=0$$
$$2s_2t_f=-3s_3{t_f}^2$$
$$s_2=\frac{-3s_3t_f}{2} \quad \text{ve} \quad s_3=-\frac{2}{t_f^3}(\theta_f-\theta_0)$$

şeklinde iki ifade elde edilir. Daha sonra $s_3=-\frac{2}{t_f^3}(\theta_f-\theta_0)$ ifadesi, (5.9) denkleminde yerine yazılarak $s_2$ katsayısı;

$$\left.\begin{matrix} s_0=\theta_0 \\ s_1=0 \end{matrix}\right\}\Rightarrow$$

$$\theta_f=s_0+s_1t_f+s_2t_f^2+s_3t_f^3$$
$$\theta_f=\theta_0+0\cdot t_f+s_2t_f^2-\frac{2s_2}{3t_f}t_f^3$$
$$\theta_f=\theta_0+s_2t_f^2-\frac{2}{3}s_2t_f^2$$
$$s_2=\frac{3}{t_f^2}(\theta_f-\theta_0) \qquad (5.12)$$

olarak bulunur. $s_2=\frac{-3s_3t_f}{2}$ ifadesi, (5.9) denkleminde yerine yazılırsa $s_3$ katsayısı;

$$\left.\begin{aligned} s_0 &= \theta_0 \\ s_1 &= 0 \end{aligned}\right\} \Rightarrow$$

$$\theta_f = s_0 + s_1 t_f + s_2 t_f^2 + s_3 t_f^3$$

$$\theta_f = \theta_0 + 0 \cdot t_f - \frac{3 s_3 t_f}{2} t_f^2 + s_3 t_f^3$$

$$\theta_f = \theta_0 - \frac{3}{2} s_3 t_f^3 + s_3 t_f^3$$

$$s_3 = -\frac{2}{t_f^3}\left(\theta_f - \theta_0\right) \qquad (5.13)$$

şeklinde elde edilir.

Bulunan bu dört katsayı aşağıdaki gibi bir arada yazılabilir [2].

$$s_0 = \theta_0$$

$$s_1 = 0$$

$$s_2 = \frac{3}{t_f^2}\left(\theta_f - \theta_0\right)$$

$$s_3 = -\frac{2}{t_f^3}\left(\theta_f - \theta_0\right)$$

# 6. YAPAY SİNİR AĞLARI

Yapay Sinir Ağları (YSA), beynin fizyolojisinden yararlanılarak oluşturulan bilgi işleme modelleridir. Literatürde yüzden fazla yapay sinir ağı modeli vardır. Bazı bilim adamları, beynin güçlü düşünme, hatırlama ve problem çözme yeteneklerini bilgisayara aktarmaya çalışmışlardır. Bazı araştırmacılar ise, beynin fonksiyonlarını kısmen yerine getiren model oluşturma konusunda çalışmalarda bulunmuşlardır [23], [27].

Yapay sinir ağlarının öğrenme özelliği, araştırmacıların dikkatini çeken en önemli özelliklerden birisidir. Çünkü herhangi bir olay hakkında girdi ve çıktılar arasındaki ilişkiyi, doğrusal olsun veya olmasın, elde bulunan mevcut örneklerden öğrenerek daha önce hiç görülmemiş olayları, önceki örneklerden çağrışım yaparak ilgili olaya çözümler üretebilme özelliği yapay sinir ağlarındaki zeki davranışın da temelini teşkil eder.

1943 yılında bir nörobiyolojist olan Warren McCulloch ve bir istatistikçi olan Walter Pitts, "*Sinir Aktivitesindeki Düşüncelere Ait Bir Mantıksal Hesap*" başlıklı bir makale ile ilk dijital bilgisayarlara ışık tutmuştur. John Von Neumann bu makaleyi, "elektronik beyinler" için bir kopya olarak görmüştür. Yapay zekâ alanındaki araştırmacılar içerisinde istisnai bir yeri olan Marvin Minsky, bu makaleden aldığı ilhamla makroskobik zekâ fikrini ortaya atmış ve uzman sistemlerin doğmasına neden olmuştur. Bronx Yüksek Bilim Okulu'ndan Frank Rosenblatt, gözün hesaplamaları ile ilgilenmiştir. Bu bilim adamları, öğrenmenin ve zekânın herhangi bir özelliğinin simülasyonunda bilgisayarların aktif olarak nasıl kullanılabileceğini, 1956 yılında düzenlemiş oldukları ilk yapay zekâ konferansında tartışmışlardır.

1959'da, Stanford üniversitesinden Bernard Widrow, basit nöron benzeri elemanlara dayanan ve "ADALINE" (Adaptive Linear Neuron) olarak adlandırılan bir adaptif lineer eleman geliştirmiştir. ADALINE ve iki tabakalı biçimi olan "MADALINE" (Multiple Adaline); ses tanıma, karakter tanıma, hava tahmini ve adaptif kontrol gibi çok çeşitli uygulamalar için kullanılmıştır. Daha sonraları ADALINE, ayrık bir çıkış yerine sürekli bir çıkış üretmek için geliştirilmiştir. Widrow, telefon hatları üzerindeki ekoları elimine etmeye yarayan adaptif filtreleri geliştirmede, adaptif lineer eleman algoritmasını kullanmıştır. Bununla ilk defa yapay sinir ağları, gerçek bir probleme uygulanmıştır.

Helsinki Teknik Üniversitesi'nden Teuvo Kohonen, 1970'lerin ilk yıllarında adaptif öğrenme ve birleşik hafızalar üzerine temel çalışmalar yapmış ve bu çalışmalar ile danışmansız öğrenme metotlarının gelişmesine ışık tutmuştur.

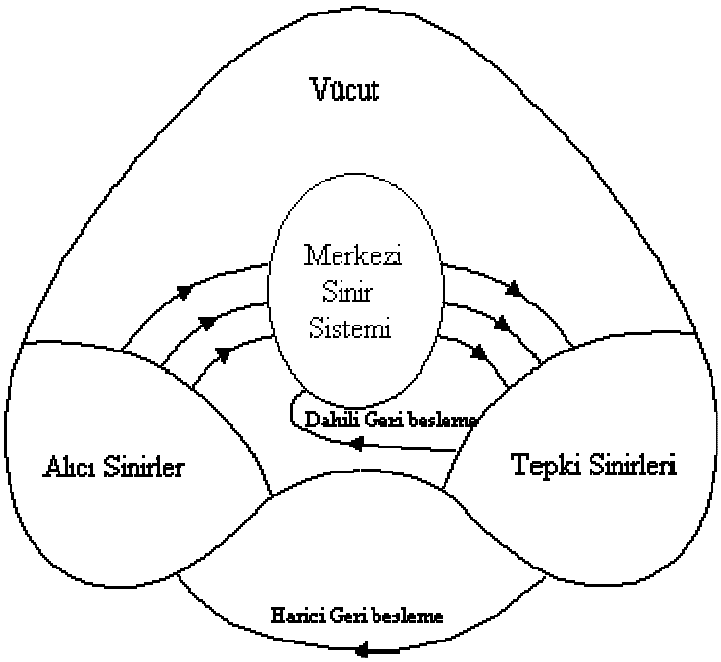
Minsky ve Papert'in Perseptron isimli kitaplarında, yapay sinir ağlarının temel olarak ilgi çekici konular olmadığını belirtmeleri, birçok araştırmacının bu alanda çalışmaktan vazgeçmelerine sebebiyet vermiştir. YSA konusunda çalışmaya devam eden Grossberg, yapay sinir ağları modellerini yapılandırmak için nörolojik verinin kullanılması, algı ve hafıza için yapay sinir ağları tabanlı mekanizmaların önerilmesi, belirgin eşitliklerle bütünleşen bir sinaptik model için, ilişkilendirici bir kural üzerinde çalışmıştır.

1982 yılında ilgi çeken bir başka gelişme, moleküler biyolojiden beyin kuramcılığına geçiş yapan bir model, Caltech fizikçisi Hopfield tarafından sunulmuştur. Kendi adıyla anılan bir ağ yapısı mevcuttur ve birçok alana uygulanmıştır.

1987 yılında yapılan ilk yapay sinir ağları sempozyumundan sonra, yapay sinir ağları uygulamaları yaygınlaşmıştır. Günümüzde, yapay sinir ağları ile ilgili araştırmalar yapan çok sayıda bilim adamı ve araştırma grupları vardır.

## *6.1. Biyolojik Nöron Yapısı*

Biyolojik sinir sistemi, merkezinde sürekli olarak bilgiyi alan, yorumlayan ve uygun bir karar üreten beynin (merkezi sinir ağı) bulunduğu üç katmanlı bir sistem olarak açıklanmaktadır. Alıcı sinirler (receptor), organizma içerisinden ya da dış ortamlardan algıladıkları uyarıları, elektriksel sinyallere dönüştürerek beyne iletirler. Tepki sinirleri (effector) ise, beyinin ürettiği elektriksel sinyalleri organizma çıktısı olarak uygun tepkilere dönüştürürler. Şekil 7.1'de biyolojik sinir sisteminin blok şeması görülmektedir [27].




**Şekil 6.1 Biyolojik sinir sisteminin blok gösterimi**

Sinir hücreleri nöron olarak bilinir. Nöron, özellikle beyin olmak üzere sinir sisteminin temel birimidir ve başlıca üç kısımdan oluşur. Bunlar:

· Gövde (cell body) ,

· Gövdeye giren sinyal alıcı lifler (dendrit),

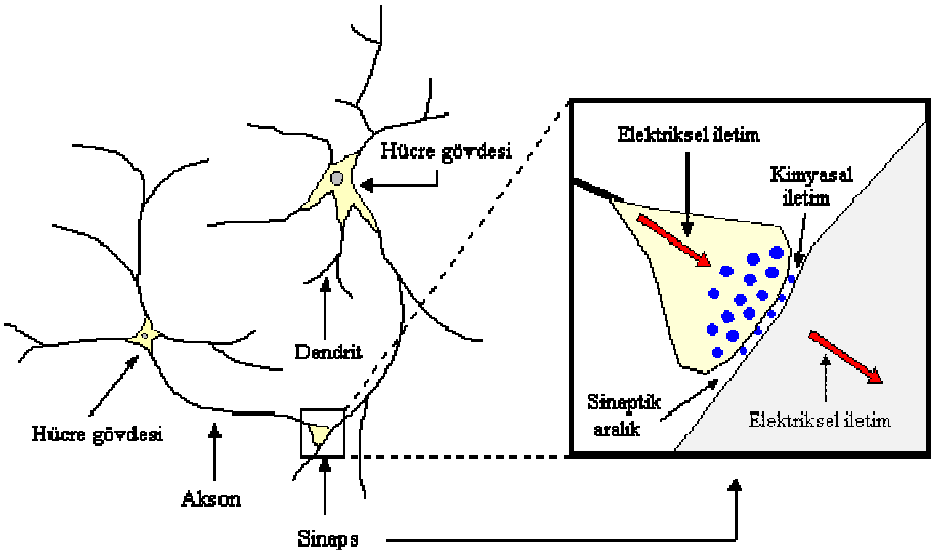
· Gövdeden çıkan sinyal iletici lifler (axon).

Yapay sinir ağları insan beyninin çalışma prensibi örnek alınarak geliştirilmeye çalışılmıştır ve aralarında yapısal olarak bazı benzerlikler vardır. Bu benzerlikler Tablo 6.1'de verilmiştir.

**Tablo 6.1 Sinir sistemi ile yapay sinir ağlarının benzerlikleri**

| SİNİR SİSTEMİ | YAPAY SİNİR AĞLARI SİSTEMİ |
|---|---|
| Neuron | İşlem elemanı |
| Dendrit | Toplama fonksiyonu |
| Hücre gövdesi | Transfer fonksiyonu |
| Aksonlar | Eleman çıkışı |
| Sinapslar | Ağırlıklar |

Şekil 6.2'de bir nöron hücresinin yapısı verilmiştir. Dendritler çevre hücrelerden gelen sinyalleri (impulse) alıp, gövdeye ulaştırırlar. Her nöron, dendritler vasıtasıyla diğer birçok nöronlardan gelen işaretleri alan ve birleştiren basit bir mikro işleme birimidir. Bir nöronun aksonu (çıkış yolu) ayrıştırılmıştır ve sinaps olarak adlandırılan bir jonksiyon vasıtasıyla diğer nöronların dendritlerine bağlanmıştır. Bu jonksiyon uçlarındaki iletim kimyasaldır ve işaretin miktarı, akson tarafından serbest bırakılan kimyasalların büyüklüğüne bağlı olarak transfer edilir ve dendritler vasıtasıyla alınır. Bu sinaptik büyüklük, beyin öğrenirken neyin modifiye edildiğini belirtir. Bu sinaps, beynin temel hafıza mekanizmasına dayanarak nöron içerisindeki bilginin işlenmesi ile birleştirilir [23], [27].

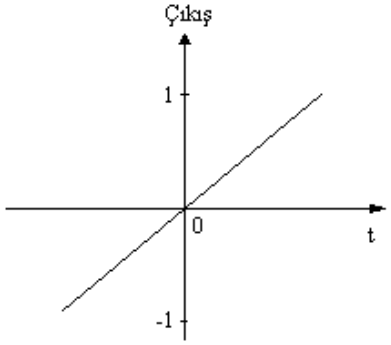



**Şekil 6.2 Biyolojik nöronun yapısı**

## *6.2. Aktivasyon Fonksiyonları*

### 6.2.1. Doğrusal ve doyumlu – doğrusal aktifleşme fonksiyonu

Hücrenin net girdisini doğrudan hücre çıkışı olarak verir. Doğrusal aktivasyon fonksiyonunun grafiği Şekil 6.3' de gösterilmiştir.

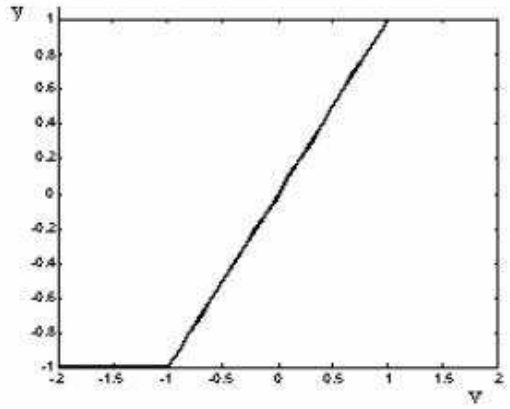



**Şekil 6.3 Doğrusal aktifleşme fonksiyonu**

Doyumlu doğrusal aktivasyon fonksiyonu ise aktif çalışma bölgesinde doğrusaldır ve hücrenin net girdisinin belirli bir değerinden sonra hücre çıkışını doyuma

götürür. Doyumlu doğrusal aktivasyon fonksiyonu (6.1) denkleminde ve grafiği Şekil 6.4' de verilmiştir [23], [27].

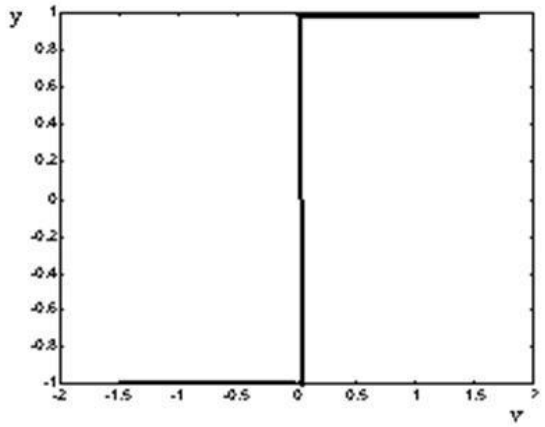
$$\begin{Bmatrix} 1 & v > 1 \\ v & -1 < v < 1 \\ -1 & v < -1 \end{Bmatrix} ise \tag{6.1}$$




**Şekil 6.4 Doyumlu doğrusal aktivasyon fonksiyonu**

## 6.2.2. Eşik sınır fonksiyonu

McCulloch-Pitts modeli olarak bilinen eşik aktifleşme fonksiyonu, mantıksal çıkış verir ve sınıflandırıcı ağlarda tercih edilir. Şekil 6.5' de grafiği gösterilmiştir [23], [27].

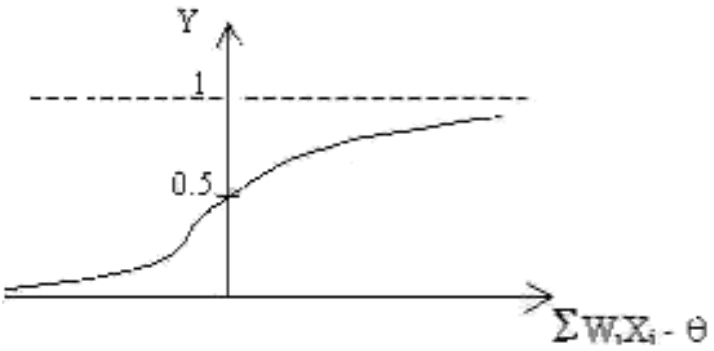



**Şekil 6.5 Eşik sınır fonksiyonu**

### 6.2.3. Sigmoid fonksiyonu

Bu fonksiyonun aktif bölgesinin, 0,2 ile 0,8 arasında olduğu bilinmektedir. Literatürde tek kutuplu aktivasyon fonksiyonu olarak da bilinir. Sigmoid fonksiyonu, türevi alınabilir, sürekli ve doğrusal olmayan bir fonksiyon olması nedeniyle doğrusal olmayan problemlerin çözümünde kullanılan YSA' nda tercih edilir. Sigmoid fonksiyonu (6.2) denkleminde ve grafiği Şekil 6.6' da verilmiştir [23], [27].

$$Y = \frac{1}{1 + e^{(-\sum X_i W_i - \theta_i)}} \tag{6.2}$$

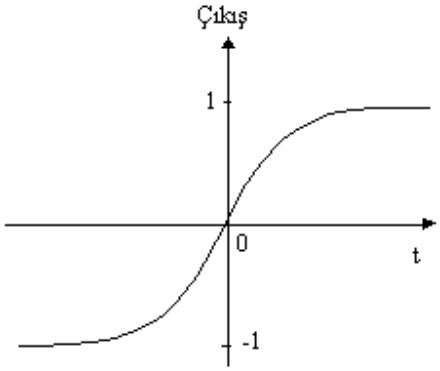



**Şekil 6.6 Sigmoid fonksiyonu**

### 6.2.4. Tanjant hiperbolik fonksiyon

Bipolar (çift kutuplu) özelliğe sahip olan tanjant hiperbolik fonksiyon, YSA uygulamalarında en çok kullanılan aktifleşme fonksiyonlarından biridir. Bu fonksiyon kullanıldığında nöronun aktifliği –1 ile 1 arasında değişir. Giriş uzayının genişletilmesinde etkili bir aktifleşme fonksiyonudur. Grafiği Şekil 6.7' de gösterilmiştir [23], [27].

Çıkış = tanh (t)




**Şekil 6.7 Tanjant hiperbolik fonksiyon**

## *6.3. İşlemci Eleman (Yapay Nöron)*

Bir YSA modelinin temel birimi, Şekil 6.8'de gösterilen işlem elemanıdır. Burada girişler dış kaynaklardan veya diğer işlem elemanlarından gelen işaretlerdir. Bu işaretler, kaynağına göre kuvvetli veya zayıf olabileceğinden ağırlıkları da farklıdır [27].

YSA'da girilen giriş değerlerine önce toplama fonksiyonları uygulanır ve her bir işlem elemanının çıkış (İEÇ) değeri;

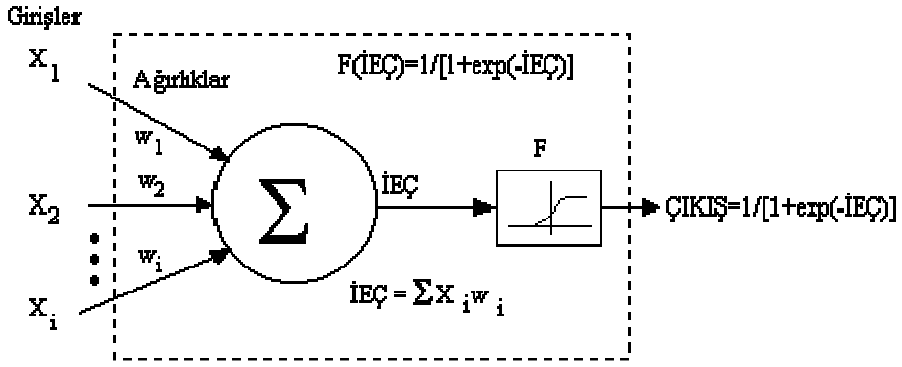
$$İEÇ = \sum_{i=1}^{N} X_i \cdot W_{ij} - \theta_i \tag{6.3}$$

olarak bulunur. Burada $X_i$ i'inci girişi, $W_{ij}$ j'inci elemandan i'inci elemana bağlantı ağırlığını ve $q_i$ eşik (threshold) değerini göstermektedir. Daha sonra bu çıkış değerleri sigmoid aktivasyon fonksiyonuna yani öğrenme eğrisine uygulanır. Sonuçta çıkış değeri;

$$ÇIKIŞ = \frac{1}{1+e^{-İEÇ}} \tag{6.4}$$

denklemi ile bulunur.

Uygulamalarda, en çok *sigmoid* veya *hiperbolik tanjant fonksiyonu* kullanılmaktadır. Şekil 6.8'de, işlemci eleman çıkışında kullanılan sigmoid fonksiyona göre çıkış değerinin hesaplanması gösterilmiştir. Bu işlemci elemanın çıkış değeri diğer işlemci elemanlarına giriş veya ağın çıkış değeri olabilir [27].




**Şekil 6.8 Bir işlemci elemanı (yapay nöron)**

## *6.4. Yapay Sinir Ağları ile Hesaplamanın Özellikleri*

Yapay sinir ağları hesaplamalardaki başarısını, paralel dağılmış yapısından, öğrenebilme ve genelleme yapma yeteneğinden alır. Genelleme, eğitim ya da öğrenme süresince kullanılmayan girişler için de yapay sinir ağlarının uygun tepkileri üretmesi olarak tanımlanır. Bu özellikleri ile yapay sinir ağları karmaşık ve çözümlenmesi güç problemleri de çözebilme yeteneğine sahiptir. Nesne tanıma, işaret işleme, sistem tanımlama ve denetimi gibi birçok mühendislik alanında yapay sinir ağları, aşağıda belirtilen özellikleri nedeniyle başarılı olmuşlardır [23], [27]:

### 6.4.1 Doğrusal olmama

Yapay sinir ağlarının temel işlem elemanı olan hücre doğrusal değildir. Dolayısıyla hücrelerin birleşmesinden meydana gelen YSA' da doğrusal değildir ve bu

özellik tüm ağa yayılmış durumdadır. Bu özelliği ile YSA, doğrusal olmayan karmaşık problemlere çözüm getirmektedir.

### 6.4.2 Öğrenme

Aslında öğrenmeden kasıt, ilgili problemdeki girdi-çıktı ilişkisini en güzel tanımlayacak optimum ağırlıkların bulunmasıdır. Problemden alınan örneklerden faydalanılarak ilgili problemi kendisine uygulanan örneklerden öğrenmeye çalışır. Probleme farklı bir çözüm sağlar.

### 6.4.3 Genelleme

Yapay sinir ağları, ilgilendiği problemi öğrendikten sonra eğitim sırasında karşılaşmadığı test örnekleri için de belirtilen tepkiyi üretme kabiliyetine sahiptir. Örneğin, karakter tanıma amacıyla eğitilmiş bir yapay sinir ağı, bozuk karakter girişlerinde de doğru karakteri verirler. Nöral hesaplamada hafızalar birleşiktir. Yani eğitilmiş ağa girişin sadece bir kısmı verilse bile, ağ hafızadan bu girişe en yakınını seçerek tam bir giriş verisi alıyormuş gibi kabul eder ve buna uygun bir çıkış değeri üretir. Veri, yapay sinir ağına, eksik, bozuk veya daha önce hiç karşılaşmadığı şekilde verilse bile, ağ kabul edilebilir en uygun çıkışı üretecektir. Bu özellik ağın genelleştirme özelliğidir.

### 6.4.4 Uyarlanabilirlik

YSA ağırlıkları, uygulanan probleme göre değiştirilir. Yani, belirli bir problemi çözmek amacıyla eğitilen yapay sinir ağı, problemdeki değişimlere göre tekrar eğitilebilir. Değişimler devamlı ise gerçek zamanda da eğitime devam edilebilir. Bu

özelliği ile yapay sinir ağları, uyarlamalı örnek tanıma, işaret işleme, sistem tanımlama ve denetim gibi alanlarda etkin olarak kullanılır.

### 6.4.5 Dağıtılmış birleşik hafıza

Yapay sinir ağlarının en önemli özelliklerinden biri de bilgiyi depolamalarıdır. Nöral hesaplamalarda bilgi ağırlıklar üzerine dağıtılmıştır. Bağlantıların ağırlıkları nöral ağın hafıza birimidir. Bu ağırlıklar ağın o andaki sahip olduğu bilgiyi veya uygulanan örneklerden öğrenmiş olduğu davranışı verir. Bu bilgiler, ağdaki birçok ağırlıklar üzerine (hafıza birimine) dağıtılır.

Eğitilmiş ağa, eğitimde kullanılmamış herhangi farklı bir giriş uygulanırsa, ağ daha önceki girişlerden öğrenmiş olduğu davranış doğrultusunda beklenen çıkışa uygun bir çıkış değeri üretebilecektir. Yapay sinir ağına uygulanan veri eksik, gürültülü veya daha önce hiç karşılaşmamış olsa bile, ağ kabul edilebilir en uygun çıkışı üretecektir. Bu özelliğe, genelleştirme özelliği denir.

### 6.4.6 Hata toleransı

Yapay sinir ağları, çok sayıda işlemci elemanların bağlantısı paralel dağılmış bir yapıya sahiptir ve ağın sahip olduğu bilgi, ağdaki tüm bağlantılara dağılmıştır. Giriş veri setinde bulunabilecek herhangi bir gürültü, bütün ağırlıklar üzerine dağıtıldığından dolayı gürültü etkisi minimize edilebilir. Geleneksel yöntemlere göre hatayı minimize etme yetenekleri daha fazladır.

### 6.4.7 Paralel işlem yapma

Yapay sinir ağları, paralel yapısı nedeniyle büyük ölçekli entegre devre (Very Large Scale Integration – VLSI) teknolojisi ile gerçeklenebilir. Bu özellik, yapay sinir

ağlarının hızlı bilgi işleme yeteneğini ve örnek tanıma, işaret işleme, sistem kimliklendirme ve denetim gibi gerçek zaman uygulamalarında kullanımını artırır.

## *6.5. Yapay Sinir Ağlarının Sınıflandırılması*

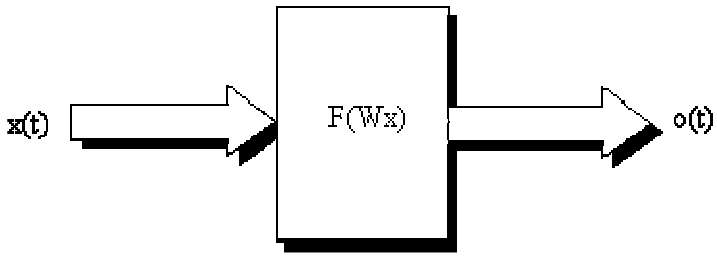
Her bir sinir hücresi arasındaki bağlantıların yapısı ağın yapısını belirler. İstenilen hedefe ulaşmak için bağlantıların nasıl değiştirileceği, öğrenme algoritması tarafından belirlenir. Kullanılan bir öğrenme kuralına göre, hatayı sıfıra indirecek şekilde ağın ağırlıkları değiştirilir. Yapay sinir ağları yapılarına ve öğrenme algoritmalarına göre sınıflandırılırlar [23], [27].

### *6.5.1. Yapay sinir ağlarının yapılarına göre sınıflandırılması*

Yapay sinir ağları, yapılarına göre, ileri beslemeli (feedforward) ve geri beslemeli (feedback) ağlar olmak üzere iki şekilde sınıflandırılırlar:

#### 6.5.1.1. İleri beslemeli ağlar

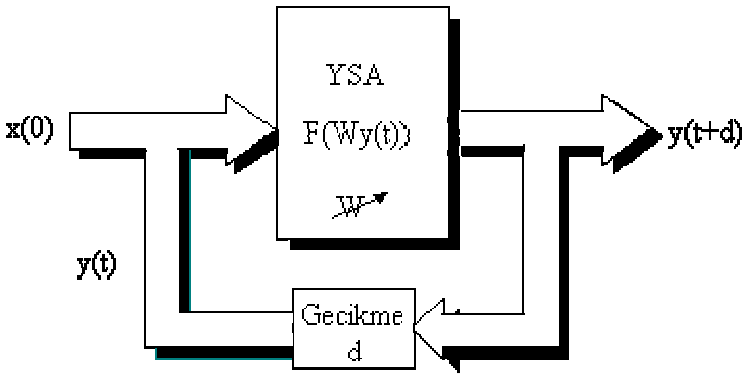
İleri beslemeli bir ağda işlemci elemanlar (İE), genellikle katmanlara ayrılmışlardır. İşaretler, giriş katmanından çıkış katmanına doğru tek yönlü bağlantılarla iletilir. İE' ler bir katmandan diğer bir katmana bağlantı kurarlarken, aynı katman içerisinde bağlantıları bulunmaz. Şekil 6.9' da ileri beslemeli ağ için blok diyagram gösterilmiştir. İleri beslemeli ağlara örnek olarak Çok Katmanlı Algılayıcı (Multi Layer Perseptron – MLP) ve Doğrusal Vektör Parçalama (Linear Vector Quantization – LVQ) ağları verilebilir.




**Şekil 6.9 İleri beslemeli ağ için blok diyagram**

### 6.5.1.2. Geri beslemeli ağlar

Şekil 6.10' da bir geri beslemeli ağ görülmektedir. Bu çeşit sinir ağlarının dinamik hafızaları vardır ve bir andaki çıkış, hem o andaki hem de önceki girişleri yansıtır. Bundan dolayı, özellikle tahmin uygulamalarında kullanılırlar. Bu ağlar çeşitli tipteki problemlerin tahmininde oldukça başarı sağlamışlardır. Bu ağlara örnek olarak Hopfield, Düzenleyici Harita (Self Organizing Map – SOM), Elman ve Jordan ağları verilebilir.



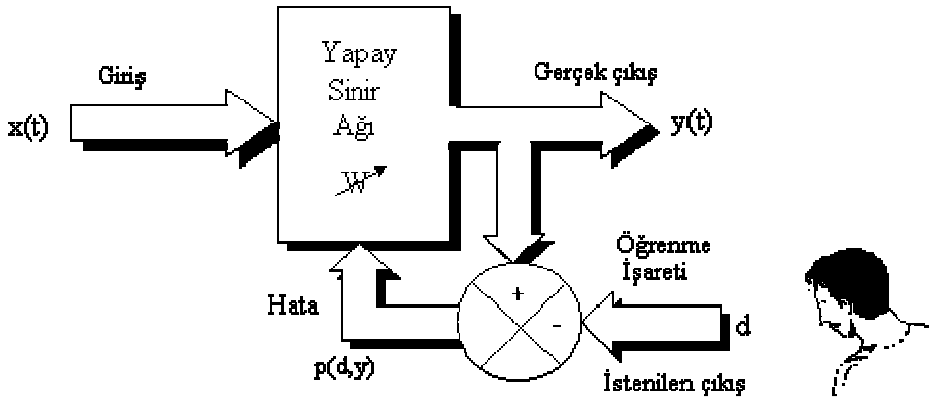

**Şekil 6.10 Geri beslemeli ağ için blok diyagram**

## 6.5.2. Yapay sinir ağlarının öğrenme algoritmalarına göre sınıflandırılması

Genel olarak üç öğrenme metodundan ve bunların uygulandığı değişik öğrenme kurallarından söz edilebilir. Bu öğrenme kuralları şu şekilde açıklanmaktadır:

### 6.5.2.1. Danışmanlı öğrenme (Supervised learning)

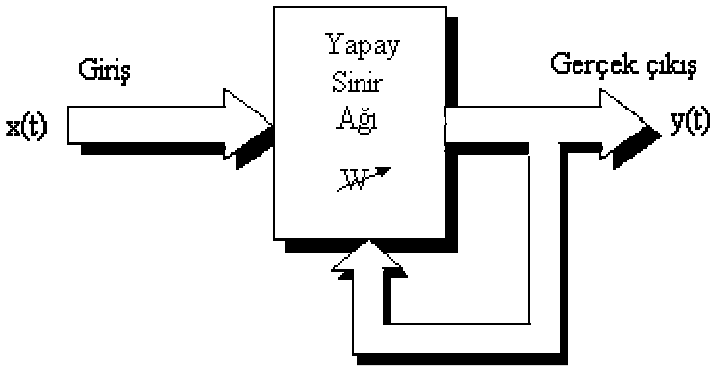
Bu tip öğrenmede, yapay sinir ağlarına örnek olarak bir doğru çıkış verilir. İstenilen ve gerçek çıktı arasındaki farka (hataya) göre İE' ler arası bağlantıların ağırlığını en uygun çıkışı elde etmek için sonradan düzenlenebilir. Bu sebeple danışmanlı öğrenme algoritmasının bir "öğretmene" veya "danışmana" ihtiyacı vardır. Widrow-Hoff tarafından geliştirilen delta kuralı ve Rumelhart ve McClelland tarafından geliştirilen genelleştirilmiş delta kuralı veya geri yayılım (back propagation) algoritması, danışmanlı öğrenme algoritmalarına örnek olarak verilebilir [27].




**Şekil 6.11 Danışmanlı öğrenme yapısı**

### 6.5.2.2. Danışmansız öğrenme (Unsupervised learning)

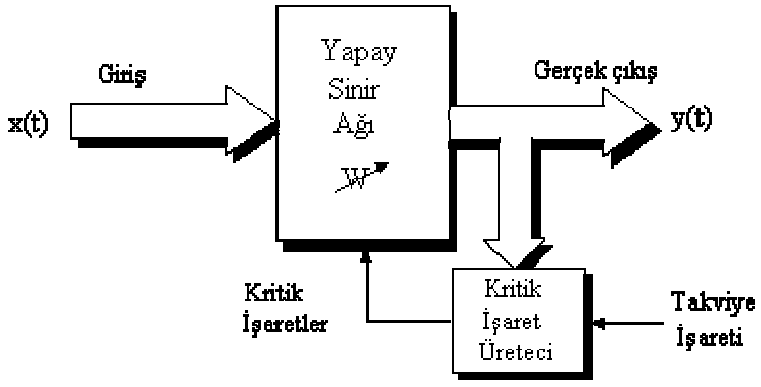
Girişe verilen örnekten elde edilen çıkış bilgisine göre ağ, sınıflandırma kurallarını kendi kendine geliştirmektedir. Bu öğrenme algoritmalarında, istenilen çıkış değerinin bilinmesine gerek yoktur. Öğrenme süresince sadece giriş bilgileri verilir. Ağ daha sonra bağlantı ağırlıklarını, aynı özellikleri gösteren desenler (patterns) oluşturmak üzere ayarlar. Grossberg tarafından geliştirilen Adaptif Rezonans Teorisi (Adaptive Resonance Theory – ART) veya Kohonen tarafından geliştirilen SOM öğrenme kuralı, danışmansız öğrenmeye örnek olarak verilebilir [27].




**Şekil 6.12 Danışmansız öğrenme yapısı**

### 6.5.2.3. Takviyeli öğrenme (Reinforcement learning)

Bu öğrenme kuralı danışmanlı öğrenmeye yakın bir metottur. Denetimsiz öğrenme algoritması, istenilen çıkışın bilinmesine gerek duymaz. Hedef çıktıyı vermek için bir "öğretmen" yerine, burada yapay sinir ağına bir çıkış verilmemekte fakat elde edilen çıkışın, verilen girişe karşılık iyiliğini değerlendiren bir kriter kullanılmaktadır. Optimizasyon problemlerini çözmek için Hinton ve Sejnowski'nin geliştirdiği Boltzmann kuralı veya genetik algoritmalar, takviyeli öğrenmeye örnek olarak verilebilirler [27].



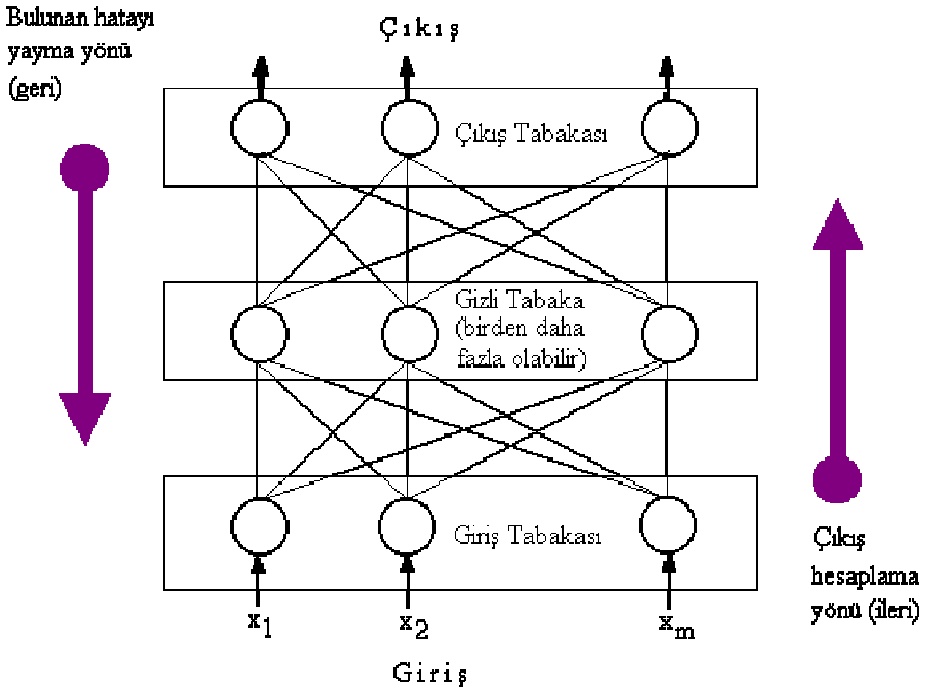

**Şekil 6.13 Takviyeli öğrenme yapısı**

## 6.6. Çok Katmanlı Algılayıcılar ve Öğrenme Algoritmaları

Çok katmanlı algılayıcılar, birçok öğretme algoritması kullanılarak eğitilebilirler. Bu çalışmada, danışmanlı öğrenme algoritmaları olan geri – yayılım (BP – backpropagation) ve takviyeli öğrenme metodu olan genetik algoritma ayrıntılı şekilde açıklanmıştır.

### 6.6.1. Çok katmanlı algılayıcılar (MLP)

Çok katmanlı bir algılayıcı (perseptron) sinir ağı modeli, Şekil 6.14' de gösterilmiştir. Bu ağ modeli özellikle mühendislik uygulamalarında en çok kullanılan sinir ağı modeli olmuştur. Bir MLP modeli, bir giriş, bir veya daha fazla ara ve bir de çıkış katmanından oluşur. Bir katmandaki bütün işlem elemanları bir üst katmandaki bütün işlem elemanlarına bağlıdır. Bilgi akışı ileri doğru olup geri besleme yoktur. Bunun için ileri beslemeli sinir ağı modeli olarak adlandırılır. Giriş katmanında herhangi bir bilgi işleme yapılmaz. Buradaki işlem elemanı sayısı, tamamen uygulanan problemin giriş sayısına bağlıdır. Ara katman sayısı ve ara katmanlardaki işlem elemanı sayısı ise, deneme-yanılma yolu ile bulunur. Çıkış katmanındaki eleman sayısı ise yine uygulanan probleme dayanılarak belirlenir [23], [27].




**Şekil 6.14 Geri yayılım çok katmanlı algılayıcı yapısı**

MLP ağlarında, ağa bir örnek gösterilir ve örnek neticesinde nasıl bir sonuç üreteceği de bildirilir (danışmanlı öğrenme). Örnekler giriş katmanına uygulanır, ara katmanlarda işlenir ve çıkış katmanından da çıkışlar elde edilir. Kullanılan eğitme algoritmasına göre, ağın çıkışı ile arzu edilen çıkış arasındaki hata geriye doğru yayılarak, hata minimuma düşünceye kadar ağın ağırlıkları değiştirilir.

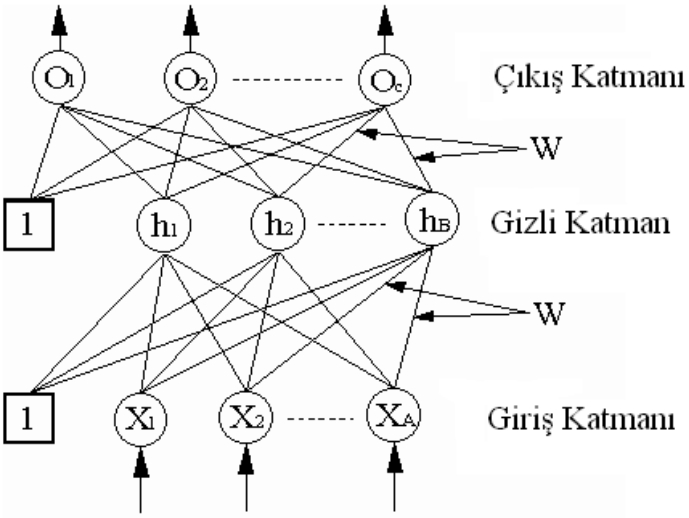
İleri beslemeli ağlar, en genel anlamıyla giriş uzayıyla çıkış uzayı arasında statik haritalama yapar. Bir andaki çıkış, sadece o andaki girişin bir fonksiyonudur [27].

### *6.6.2. Geri yayılım algoritması*

Giriş, çıkış ve en az bir tane gizli katman olmak üzere üç katmandan oluşmaktadır. Her katmandaki düğümler, kendisinden bir önceki ve sonraki katmandaki tüm düğümlere bağlıdırlar [23], [27].

Geriye yayılma algoritmasının adımları:

Her bir $w_i$ başlangıç ağırlıkları için küçük değerler verilir.




**Şekil 6.15 Geri yayılım algoritması mimarisi**

Sonlandırma şartı sağlanana kadar, her eğitim örneği için <($x_1$, $x_2$,.....$x_n$)t>

Giriş örnekleri ($x_1$, $x_2$,.....$x_n$) ağa alınır ve aktifleşmeler hesaplanır.

Gizli katmandaki her bir nöron için aktifleşme $F_h$ hesaplanır.

$$net_i = \sum_{i=}^{n} F_i w_{ij} \tag{6.5}$$

$$F_h = \frac{1}{1 + e^{(-netj)}} \tag{6.6}$$

Çıkış katmanındaki her bir nöron için aktifleşme $F_j$ hesaplanır.

$$net_j = \sum_{h}^{n} F_h W_{hj} \tag{6.7}$$

$$F_j = \frac{1}{1 + e^{(-net_j)}} \tag{6.8}$$

Hata değerleri hesaplanır.

Toplam hata değeri hesaplanır:

$$TH = \frac{1}{2}\sum_{m} E_m^2 \tag{6.9}$$

Her çıkış nöronu için geriye yayılma hatası $\delta_j$ hesaplanır.

$$\delta_j = F_j\left(1 - F_j\right)\cdot\left(T_j - F_j\right) \tag{6.10}$$

Bu adımda, önce verilmiş bütün çıkış hedefleri ile elde edilmiş çıkış değerleri kıyaslanır. Eğer bu değerler farklı ise hesaplama devam eder. Farklı değilse hesaplama bitmiştir.

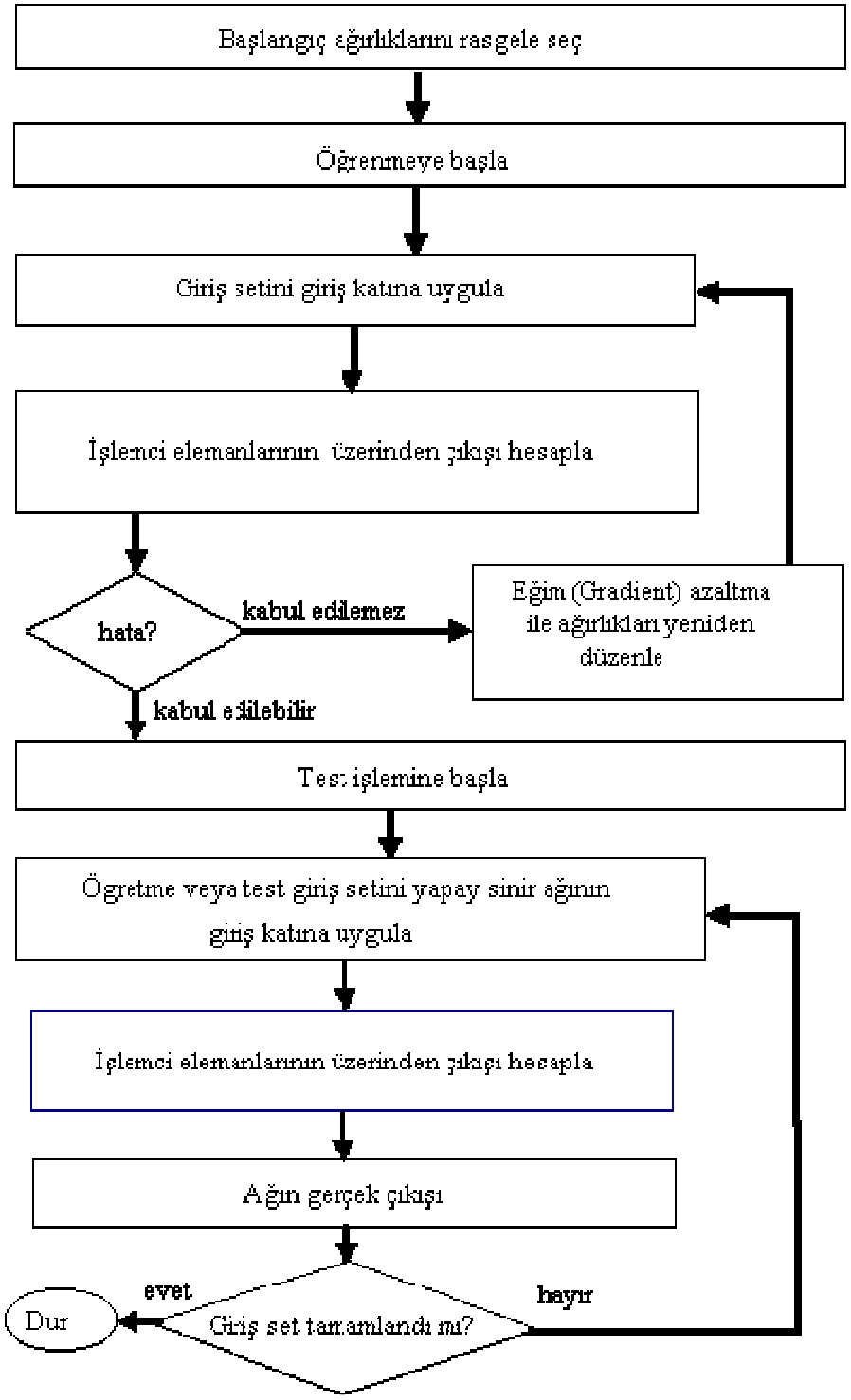
Gizli katmanlardan en son katmandaki nöronlar için geriye yayılma hatası $\delta_h$ hesaplanır.

$$\delta_h = F_h\left(1 - F_h\right)\sum_{j} \delta_j W_{hj} \tag{6.11}$$

Ağırlık değerleri $w_{ij}$ güncelleştirilir.

$$w_{ji}\left(t+1\right) = w_{ji}\left(t\right) + \Delta w_{ji} \qquad \Delta w_{ji} = \alpha \cdot \delta_i \cdot F_{ji} \tag{6.12}$$

Yeni $w_{ji}$ ağırlıkları (6.12) denklemi ile elde edilir.




**Şekil 6.16 Çok katmanlı algılayıcının geri yayılım akış şeması**

## *6.7. Yapay Sinir Ağlarının Avantajları ve Dezavantajları*

**Avantajları;**

Yapay sinir ağları ile matematiksel olarak modellenmesi zor veya imkansız olan problemler rahatlıkla çözülebilmektedir.

Gerçek dünyada yaşana olaylar arasındaki karmaşık ilişkilerin belirlenim sunulması zordur. Ancak yapay sinir ağı dışarıdan bu şekilde bir desteğe ihtiyaç duymadan, örneklerden öğrenir.

Geleneksel yöntemlerde olaylar arasındaki ilişki doğrusal değilse modellenmesi zordur ya da bazı varsayımlar gerektirir. Bu varsayımlar modellenen sistem ile gerçek sistem arasında fark oluşturacağından elde edilen sonucun gerçeği yansıtmama ihtimali yüksektir. Yapay sinir ağları eğitiminde örnekler arasındaki ilişkilerin doğrusal olması gerekmemektedir. Sistem verilen örnekler üzerinden öğrenerek ilişkiyi kendisi kurar. Modelin gerçek sisteme daha yakın olması sebebi ile sonuç gerçek sonuca çok yakın olmaktadır.

YSA uygulamaları, hem pratik hem de maliyet bakımından ucuzdurlar. Örneklerin belirlenmesi ve bir program, problemin çözümü için yeterli olmaktadır.

YSA' ların paralel çalışabilme özellikleri gerçek zamanlı kullanımlarını kolaylaştırmaktadır.

Yapay sinir ağları, yeni bilgilerin ortaya çıkması veya ağdaki bazı bilgilerin değişmesi durumunda yeniden eğitilebilmektedirler [23].

**Dezavantajları;**

Ağın oluşturulmasında, model seçilmesinde, ağın topolojisinin belirlenmesinde herhangi bir kural yoktur. Kullanıcı tecrübe ederek en uygun ağı oluşturur.

Aynı problem değişik şekillerde gösterilince, her gösterimin kendisine göre performansı değişmektedir. Örneklerin ağa doğru şekilde gösterilmesi tecrübeye bağlıdır.

Ağın davranışının açıklanması mümkün değildir. Bu sebeple ağa olan güven azalmakta ve özellikle insan hayatı ile ilgili problemlerde, sonuçların neden verilmediğinin açıklanamaması, kullanım alanlarını sınırlandırmaktadır.

YSA, probleme optimum sonuç üretileceğini garanti edemez. Bulunan sonuç ancak iyi sonuçlardan birisidir denilebilir. Geleneksel yöntemler daha çok zaman harcamalarına rağmen optimum sonuç üretirler.

Problemi temsil edecek örneklerin zor bulunduğu veya bulunamadığı durumlarda, problem için kabul edilebilir çözümler üretmek zorlaşmakta veya mümkün olmamaktadır.

Eğitimin gerçekleştirilmesi çok uzun zaman alabilmektedir [23].

## *6.8. Genetik Algoritma*

### 6.8.1. Genetik algoritmanın tanımı

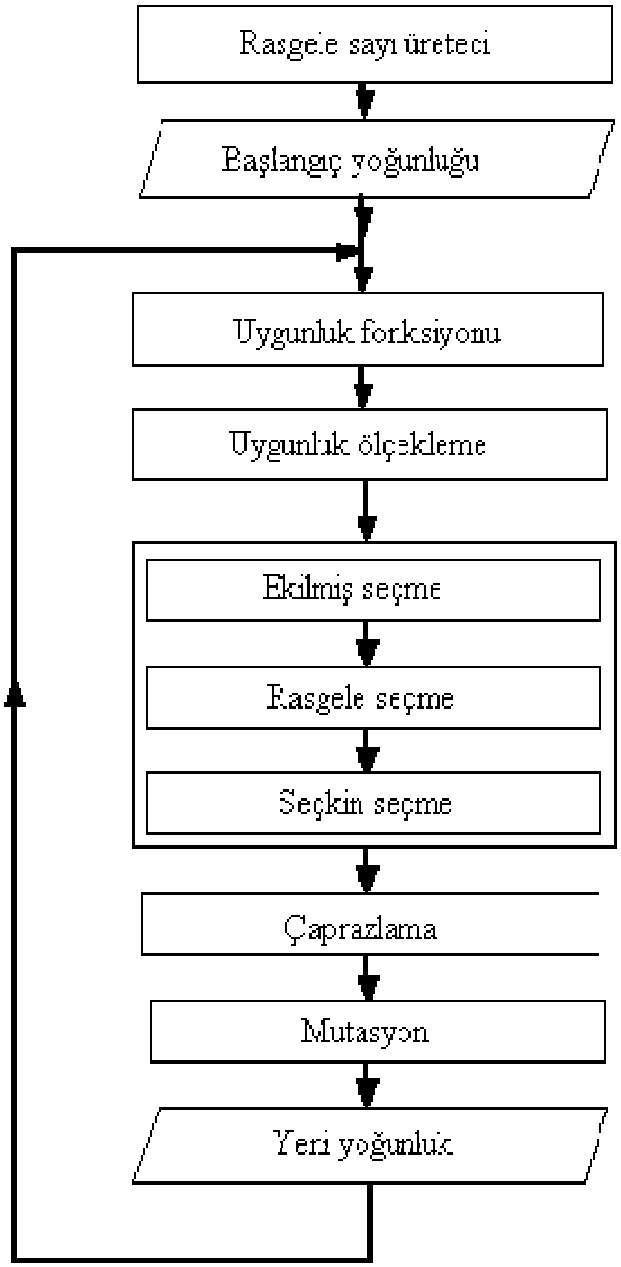
Genetik algoritmalar (GA), yapay zekânın gittikçe genişleyen bir kolu olan evrimsel hesaplama tekniğinin bir parçasını oluşturmaktadır. Darwin'in evrim teorisinden esinlenerek oluşturulmuştur. Bir problemin genetik algoritma ile çözümü, problemi sanal olarak evrimden geçirmek suretiyle yapılmaktadır [11], [30].

Evrimsel hesaplama ilk olarak 1960' larda I. Rechenberg tarafından "Evrim Stratejileri (Evolutions Strategie)" isimli eserinde tanıtılmıştır. John Holland, evrim sürecinin matematiksel modeli kullanılarak bilgisayarın, karmaşık problemleri öğrenip çözüm geliştirebileceğini düşünmüştür. Böylece bu düşüncenin sonucu olarak GA bulunmuştur. Onun öğrencileri ve arkadaşları tarafından geliştirilmiş ve Holland'ın kitabı "Doğal ve Yapay Sistemlerde Adaptasyon (Adaption in Natural and Artificial Systems)" 1975 yılında yayınlanmıştır.

1992 yılında John Koza, genetik algoritmayı kullanarak çeşitli görevleri yerine getiren programlar geliştirerek bu metoda Genetik Programlama adını vermiştir. Genetik algoritma, geleneksel yöntemlerle çözümü zor veya imkânsız olan problemlerin çözümünde kullanılmaktadır. Çok genel anlamda genetik algoritmanın üç uygulama alanı bulunmaktadır. Bunlar deneysel çalışmalarda optimizasyon, pratik endüstriyel uygulamalar ve sınıflandırma sistemleridir.

GA, mühendislik problemlerinde optimizasyon amaçlı olarak kullanılmaya başlanmıştır. Özellikle mekanizma tasarımında çok iyi sonuçlar verdiği bilinmektedir. Bunlardan başka otomatik programlama, öğrenme kabiliyetli makineler, ekonomi, çevrebilim, planlama, üretim hattı yerleşimi gibi alanlarda da uygulanmaktadır. Ayrıca dijital resim işleme tekniğinde de çokça uygulama alanı bulmuştur.

Bu problemlerin hemen hepsi çok geniş bir çözüm havzasının taranmasını gerektirmektedir. Bu çözüm havzasının geleneksel yöntemlerle taranması çok uzun sürmekte, genetik algoritmayla ise kısa bir sürede kabul edilebilir bir sonuç alınabilmektedir [30].



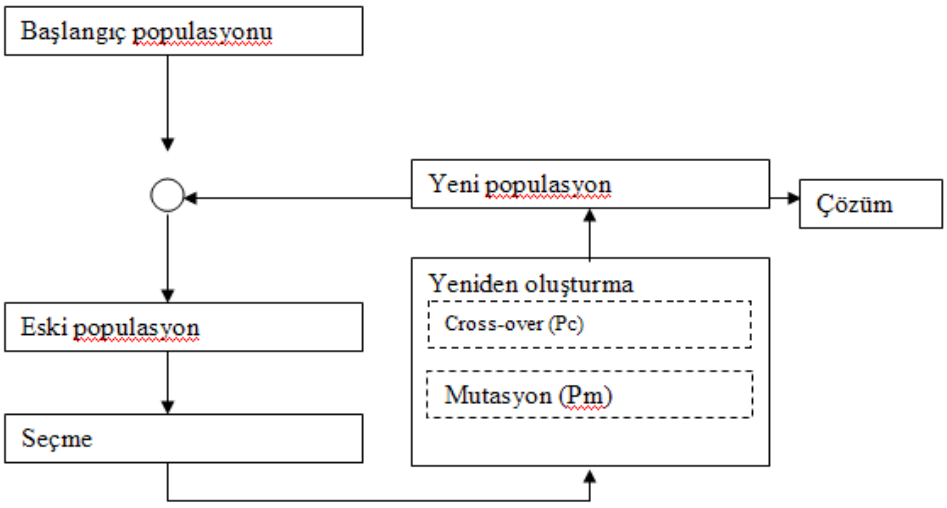

**Şekil 6.17 Genetik algoritmanın akış şeması**

### 6.8.2. Genetik algoritmaların çalışma prensibi

Genetik algoritmanın çalışması aşağıdaki şekilde özetlenebilir:

**Tablo 6.2 Genetik algoritmaların çalışma şekli**

| | |
|---|---|
| Adım 1 | Olası çözümlerin kodlandığı bir çözüm grubu oluşturulur (çözüm grubu, biyolojideki benzerliği nedeniyle, toplum (population), çözümlerin kodları (string) da kromozom olarak adlandırılır). |
| Adım 2 | Her kromozomun ne kadar iyi olduğu bulunur (fitness function). |
| Adım 3 | Bu kromozomlar eşlenerek (mating), yeniden kopyalama (recombination) ve değiştirme (crossover) operatörleri uygulanır. Bu sayede yeni bir toplum oluşturulur. |
| Adım 4 | Yeni kromozomlara yer açmak için eski kromozomlar ortadan kaldırılır. |
| Adım 5 | Tüm kromozomların uygunlukları tekrar hesaplanır. |
| Adım 6 | Eğer jenerasyon süresi dolmamışsa 3. adıma gidilir. |
| Adım 7 | O ana kadar bulunmuş en iyi kromozom sonuçtur. |




**Şekil 6.18 Genetik algoritmanın temeli**

Genetik algoritmanın yapısı oldukça geneldir ve herhangi bir probleme uygulanabilir. Kromozomların tanımlanması genellikle ikili düzendeki sayılarla yapılır. Çaprazlama işlemi için kullanılan bireyler iyi bireylerden seçilir.

GA kullanılarak bir problem çözülecekse, algoritmanın ne zaman sonlanacağına kullanıcı karar vermektedir. GA' nın belli bir sonlanma ölçütü yoktur. Sonucun yeterince iyi olması veya yakınsamanın sağlanması algoritmanın durması için ölçüt olarak kullanılabilir [11], [13], [30].

### 6.8.3. Genetik algoritmalarda kullanılan operatörler

Genetik algoritmanın en önemli kısımları çaprazlama ve mutasyon işlemleridir. Bu işlemler bir olasılık değeri ile ve genelde rastgele olarak uygulanır. Bu şekilde iyi sonuç alınabilmektedir [4].

Bir kromozomun ikili sayılarla temsil edilmesi:

*Kromozom 1* `1101100100110110`

*Kromozom 2* `1101111000011110`

Kromozom, temsil ettiği çözümle ilgili bilgi içermelidir. Her kromozom ikili (binary) bir diziden oluşur ancak bunun yerine tamsayı veya reel sayılar da kullanılabilir. Bu dizi içindeki bit adı verilen her bir sayı, çözümün bir karakteristiğini temsil edebilir veya bir dizi bütünüyle bir sayıya işaret edebilir [11], [30].

#### *6.8.3.1 Üreme:*

Üreme işlemi, belli bir seçme kriterine göre bireylerin seçilip yeni kuşağın oluşturulması işlemidir. Seçme kriterleri uyumluluğu esas alarak, birbiriyle uyumlu olan bireyleri seçer.

#### *6.8.3.2 Çaprazlama:*

Çaprazlama, ebeveynlerden bazı genleri alarak yeni bireyler oluşturma işlemidir.

*Kromozom 1* `11011 | 00100110110`

*Kromozom 2* `11011 | 11000011110`

*Birey 1* `11011 | 11000011110`

*Birey 2* `11011 | 00100110110`

Çaprazlama yapılacak konum rastgele seçilir. Oluşan yeni birey ebeveynlerin bazı özelliklerini almış ve bir bakıma ikisinin kopyası olmuştur. Birden fazla çaprazlama noktası da seçilebilir.

#### *6.8.3.3 Mutasyon:*

Oluşan yeni çözümlerin önceki çözümü kopyalamasını önlemek ve sonuca daha hızlı ulaşmak amacıyla yapılır. Mutasyon oluşan yeni bireyin bir bitini rastgele değiştirir.

*Orijinal Birey 1* `1101111000011110`

*Değişmiş Birey 1* `1100111000011110`

*Orijinal Birey 2* `1101100100110110`

*Değişmiş Birey 2* `1101101100110110`

#### *6.8.3.4 Elitizm:*

Üreme, çaprazlama ve mutasyon işlemleri sonrasında kuşakta bulunan en iyi uyumluluğa sahip birey, bir sonraki kuşağa aktarılamayabilir. Bunu önlemek için, bu işlemlerden sonra oluşan yeni kuşaktaki herhangi bir birey ile, bir önceki kuşağın en iyi (elit) bireyi değiştirilir.

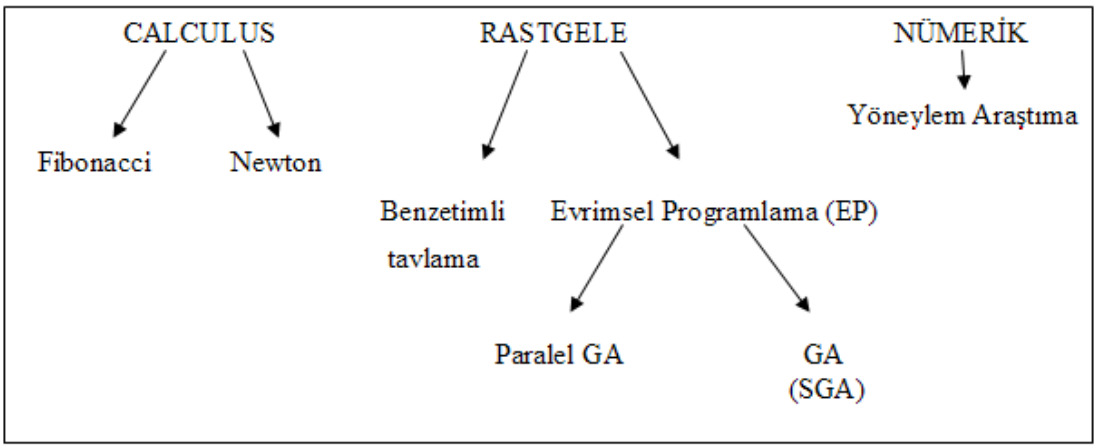
### 6.8.4. Genetik algoritmaların kullanılma nedenleri

Diğer yöntemlerin neden kullanılmadığı belirtilerek genetik algoritmaların neden tercih edildiği daha iyi anlaşılabilir.

Denklem en iyilemesinde (optimization) ;

1. Türev-İntegral hesabına (calculus) dayananlar,
2. Rastgele aramalar (random searches),
3. Numaralamaya (enumeration) dayananlar,

olmak üzere üç tip çözümden bahsedilir. Genetik algoritmaların yeri Şekil 6.19' da şematik olarak verilmiştir.




**Şekil 6.19 Denklem en iyileme yöntemleri**

Türev–İntegral hesaplamalarına dayanan hesaplama yöntemleri çok derinlemesine çalışılmıştır. Bu yöntemler fonksiyonun türevinin köklerinin, fonksiyonun en küçük ve en büyük değer veren noktaları olmasından yararlanır. Gerçek problemler için sıfır veren noktaları bulmak da ayrı bir problemdir.

Diğer bir yöntem ise, alınan bir noktadan sadece yukarı ilerleyerek en iyi sonucu bulmayı hedefler. *Tepe tırmanma* (hill climbing) denen bu yöntem, fonksiyon grafiğinin tepelerini tırmanır. Ancak çok sayıda dönme noktası içeren bir fonksiyonda, çok sayıda

tepe oluşur. Hangi tepenin en iyi çözüm olduğu bilenemez. Numaralama yöntemleri ise oldukça alışılagelmiştir. Sürekli olan gerçel sayı aralıkları, belli sayıda parçaya ayrılarak parçalar denenir. Bu yöntemin biraz daha geliştirilmiş şekli dinamik programlamayla (dynamic programming) oluşturulur. Parçalar arasından iyi görünenler seçilir. Bu parçalar, parçalara ayrılarak işlem tekrarlanır. Bu yöntem de tepe tırmanma yöntemi gibi yanlış tepeleri araştırabilir. Dinamik programlama, tepelerin olmadığı aralıklarda başarılı ve hızlıdır.

En iyilemenin;

1. Bir işin daha iyi yapılması,
2. En doğru şekilde yapılması

olmak üzere iki amacı vardır.

Günümüzde, rastgele aramaların kullanımı artmaktadır. Bu tip aramalar, en iyilemenin daha iyi yapma amacını sağlamakta daha başarılıdırlar [11], [30], [37].

### *6.8.5.* Genetik algoritmaların farkları

Genetik algoritma parametrelerin kodlarıyla uğraşır. Parametreler kodlanabildiği sürece fark etmez.

Genetik algoritma bir tek yerden değil, bir grup çözüm içinden arama yapar.

Genetik algoritma ne yaptığı konusunda bilgi içermez, nasıl yaptığını bilir. Bu nedenle bir kör arama (blind search) metodudur.

Genetik algoritmalar olasılık kurallarına göre çalışır. Programın ne kadar iyi çalışacağı önceden kesin olarak belirlenemez. Ama olasılıkla hesaplanabilir.

Genetik algoritmaların işleyişi çok basittir, fakat bu kadar basit olan yöntemin, en zor problemleri nasıl çözdüğünün anlaşılması da o kadar zordur. Bu da GA' nın en karmaşık ve bilim adamlarının yıllardır çözmeye çalıştıkları en önemli sorularından biridir. GA, çözüm/çözümler bulmak için taranması gereken parametre uzayının çok

büyük olduğu durumlarda, arama işlemi için en kullanışlı yöntemdir. Evrimin her sürecinde edinilen bilgi sonraki nesillere aktarılarak taramanın daha uygun bölgelerde gezmesi sağlandığı gibi, değişim işlemi yardımıyla yerel çözüm noktalarına sıkışıp kalma olasılığı da azaltılır.

Genetik algoritmanın paralel işlem yapılan bilgisayarlarda kullanılmaya elverişli yapısı da zaman alıcı problemlerin çözümü için uygun bir seçenek olmasını sağlamaktadır [11], [15], [30], [32].

### 6.8.6. Genetik algoritmanın performansını etkileyen nedenler

1. *Kromozom sayısı*: Kromozom sayısını arttırmak çalışma zamanını arttırırken azaltmak da kromozom çeşitliliğini yok eder.

2. *Mutasyon Oranı*: Kromozomlar birbirine benzemeye başladığında hala çözüm noktalarının uzağında bulunuyorsa, mutasyon işlemi GA' nın sıkıştığı yerden kurtulmak için tek yoludur. Ancak yüksek bir değer vermek, GA' yı kararlı bir noktaya ulaşmaktan alıkoyacaktır.

3. *Kaç Noktalı Çaprazlama Yapılacağı*: Normal olarak çaprazlama tek noktada gerçekleştirilmekle beraber yapılan araştırmalar bazı problemlerde çok noktalı çaprazlamanın çok yararlı olduğunu göstermiştir.

4. *Çaprazlamanın sonucu elde edilen bireylerin nasıl değerlendirileceği*: Elde edilen iki bireyin birden kullanılıp kullanılamayacağı bazen önemli olmaktadır.

5. *Nesillerin birbirinden ayrık olup olmadığı*: Normal olarak her nesil, tümüyle bir önceki nesle bağlı olarak yaratılır. Bazı durumlarda yeni nesli eski nesille birlikte, yeni neslin o ana kadar elde edilen bireyleri ile yaratmak yararlı olabilir.

6. *Parametre kodlanmasının nasıl yapıldığı*: Kodlananın nasıl yapıldığı en önemli noktalardan biridir. Örnek vermek gerekirse kimi zaman bir parametrenin doğrusal yada logaritmik kodlanması GA' nın performansında önemli bir farka yol açabilir.

7. *Kodlama gösteriminin nasıl yapıldığı*: Bu da nasıl olduğu yeterince açık olmamakla beraber, GA' nın performansını etkileyen bir noktadır. İkilik düzen, kayan nokta aritmetiği ya da gri kodlama ile gösterim en yaygın yöntemlerdir.

8. *Başarı değerlendirmesinin nasıl yapıldığı*: Akıllıca yazılmamış bir değerlendirme işlevi, çalışma zamanını uzatabileceği gibi çözüme hiçbir zaman ulaşmamasına neden olabilir [11], [13], [30].

### 6.8.7. Uygulama alanları

Genetik algoritmaların en uygun olduğu problemler, geleneksel yöntemler ile çözümü mümkün olamayan ya da çözüm süresi problemin büyüklüğü ile üstel orantılı olarak artanlardır. Bugüne kadar GA ile çözümüne çalışılan konulardan bazıları aşağıda tanımlanmıştır:

1. *Optimizasyon*: Sayısal optimizasyon ve kombinetoral optimizasyon problemleri olan devre tasarımı, doğrusal olmayan denklem sistemlerinin çözümünde ve fabrika-üretim planlamasında kullanılır.

2. *Otomatik Programlama (automatic programming)*: Bilgisayar programları yardımıyla network sıralamasında (sorting), ders programı hazırlanmasında kullanılır.

3. *Makine öğrenmesi (machine learning)*: Robot sensörlerinde, YSA' larda, VLSI yonga tasarımı ve protein yapısal analizinde kullanır.

4. *Ekonomi (economics)*: Ekonomik modellerin geliştirilmesinde ve işlemesinde kullanılır.

5. *İmmün sistemler (Immune systems)*: Çok-gen' li ailelerin evrimi esnasında ve doğal immün sistem modellerinde kullanılır.

6. *Popülasyon genetiği (population genetics)*: Evrim ile ilgili sorulara cevap bulmada kullanılır.

7. *Evrim ve öğrenme (evolution and learning)*: Fertlerin öğrenmesini ve türlerin evrilmesinde kullanılır.

8. *Sosyal sistemler (social systems)*: Sosyal sistemlerin analizinde kullanılır [13], [30].

### 6.8.8. Genetik algoritmalar için değerlendirme

Genetik algoritma çalışmaya başladıktan bir süre sonra, en anlamlı basamaklarda bir değer baskın gelir. Bir basamakta bir değerin baskın gelmesi, ilgili noktanın genetik algoritmanın araştırma işleminde yer almaması anlamına gelir. Sonuç olarak, o noktadaki değerin araştırması bitmiş sayılabilir. Genetik algoritma bu aşamadan sonra, o nokta hiç yokmuş gibi diğer noktalarla işleme devam eder. Bu aşamadan sonra genetik algoritmanın daha başarılı bir değer bulması daha zordur. Benzerlikler fazla olduğu için arama uzayının değişik bölgelerine ulaşılamaz. Toplum bu duruma ulaştığında en genel olan çözüm, en iyi çözüm olmak zorunda değildir. Genetik algoritma, bu genel çözüme ulaşırken daha iyi çözümler bulmuş olabilir.

Bit pozisyonlarının frekansları, yukarıda belirtilen olaya bağlı olarak sırayla baskın hale geçer. Genetik algoritmada gözlemlenen bu durumun bazı istisnaları vardır:

- Bazı basamaklar normalden daha erken baskın duruma geçer.
- Bazı basamaklar baskın değerlerini değiştirirler.
- Bazı basamaklar hiçbir zaman baskın hale geçmez.

Genetik algoritmanın başarısı kısaca problemi parçalayarak çalışmasıyla açıklanabilir. Genetik algoritmanın bu yöntemlere göre daha başarılı olmasının en önemli nedeni, problemin boyutundan bağımsız olmasıdır. Değişken sayısı ve bu değişkenlerin değerlerinin sonuçlara etkisi, araştırma yöntemini değiştirmez. Araştırma, arama uzayını dinamik olarak parçalar. Algoritmanın çalışması sırasında çevre etkenlerine bağlı olarak arama uzayı farklı şekilde taranır. En alt düzeyde işlemler

oldukça karmaşık ve değişkendir; ancak üst düzey bir bakışla değişik yollarla da olsa genetik algoritma sonuca ulaşır.

Genetik algoritmalar ayrıca bazı geri dönüşler içerir. Bu geri dönüşler sayesinde daha karmaşık problemler çözülebilir ve genetik algoritma yerel maksimum noktalara takılmaktan kurtulur. Bu geri dönüşlere, genetik algoritmanın içinde bulunan çeşitlilik neden olur. Baskın değerin yanında bazı farklı değerler de denenmeye devam eder. Kimi durumlarda bu farklı değerler baskın değerden daha başarılı olur. Bu olay genetik algoritmanın frekans grafiklerinde baskın değerini değiştiren noktalar olarak kendini gösterir [11], [15], [30], [32], [37].

## *6.9. Diğer Çok Katmanlı Perseptronlar*

1. Delta – Bar – Delta
2. Genişletilmiş Delta – Bar – Delta (Extended DBD)
3. Hızlı Yayılım (QuickProp)

## *6.10. Diğer Yapay Sinir Ağları*

1. LVQ (Learning Vector Quantization)
2. Hopfield ağı
3. Elman ve Jordan Ağları
4. Kohonen Ağı
5. ART (Adaptive Resonance Theory) ağı

# 7. ENGELLİ ORTAMDA ÇALIŞAN ROBOTUN YÖRÜNGE PLANLAMASI İÇİN YAPAY SİNİR AĞLARI UYGULAMASI

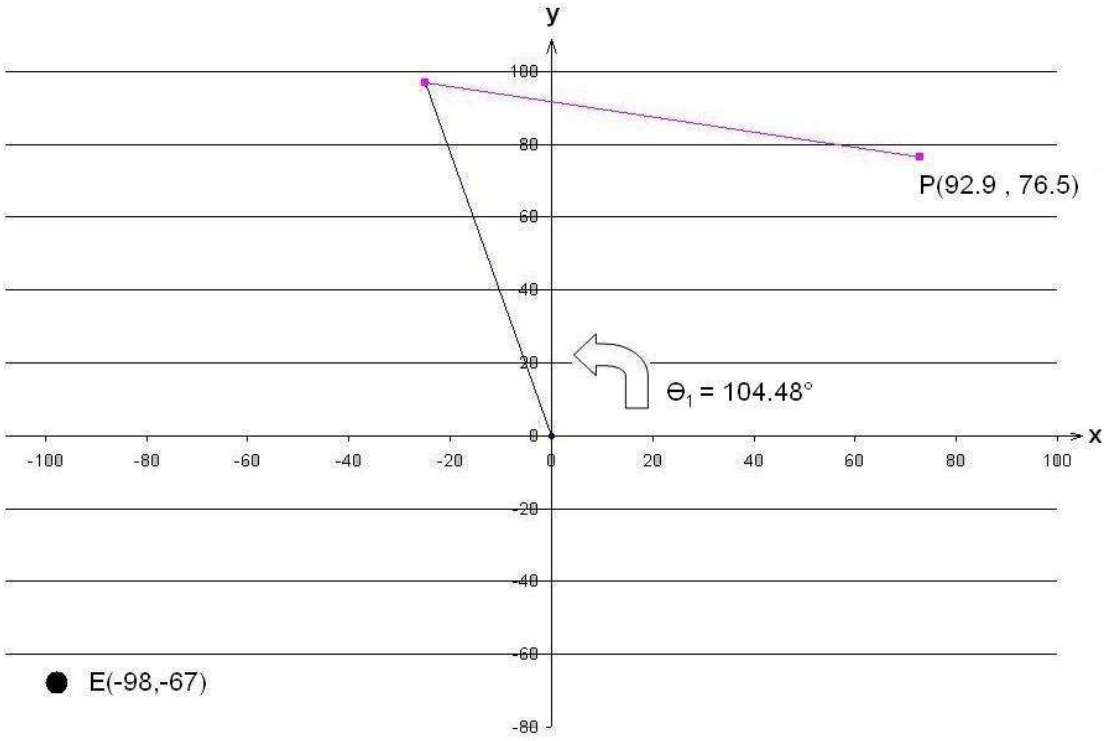
Uygulamanın amacı, ters kinematik analiz ile bulunan $\theta_1$ ve $\theta_2$ değerlerini, YSA ve GA kullanarak yeniden elde etmektir.

Ters kinematik hesaplamalar, Office Excel programı kullanılarak gerçekleştirilmiştir. Robot kolun ulaşması istenilen hedef nokta belirlenmiş, bu noktaya ulaşmak için gerekli olan $\theta_1$, $\theta_2$ ve $d_3$ değerlerini matematiksel olarak otomatik hesaplayacak Excel tablosu oluşturulmuştur. Hedef nokta için verilecek olan x, y ve z koordinat değerleri için rastgele olacak şekilde, 2000 adet örnek oluşturulmuştur. Bu örnekler Excel' de alt alta yazılarak hesaplanmış, 2000 adet $\theta_1$, $\theta_2$ ve $d_3$ değeri elde edilmiştir.

Robotun kaçınması gereken engel koordinatları, iki boyutlu olarak (x,y) verilmiştir. z koordinat değeri, $(-\infty, +\infty)$ aralığında düşünülmüştür. Yani, robotun çalışma alanına dik, robotun altından veya üsten geçemeyeceği kadar uzun bir çubuk, engel olarak öngörülmüştür. Bu engel için verilen (x, y) koordinat değerleri, yine Excel programı kullanılarak, 2000 adet olacak şekilde rastgele oluşturulmuştur.

Matematiksel olarak yapılan hesaplamaların sonucu ve engelin koordinatları ile, Excel kullanılarak, her hesaplama için ayrı ayrı olmak üzere toplam 2000 adet grafik çizilmiştir. Bu grafikler, robot kolunun üstten görünüşü olarak çizdirilmiş ve engel koordinatı da noktasal olarak grafik üzerinde gösterilmiştir. Hesaplanan $\theta_1$ ve $\theta_2$ değerleri her grafik için gözlenmiş, robot kolunun dönüş yönü belirlenmiş, bu dönüş yönüne ve engel koordinatına göre, çarpışma yoksa bu açılar kabul edilmiş, ancak çarpışma varsa, birinci kol ters yönde döndürülerek çarpışmanın olmaması sağlanmıştır. Bu şekilde, çarpışmalı ve çarpışmasız olan durumlarda kullanılacak yeni $\theta_1$ değerleri

oluşturulmuştur. Oluşturulan grafiklerden üç tanesi, Şekil 7.1, Şekil 7.2 ve Şekil 7.3' de gösterilmiştir.



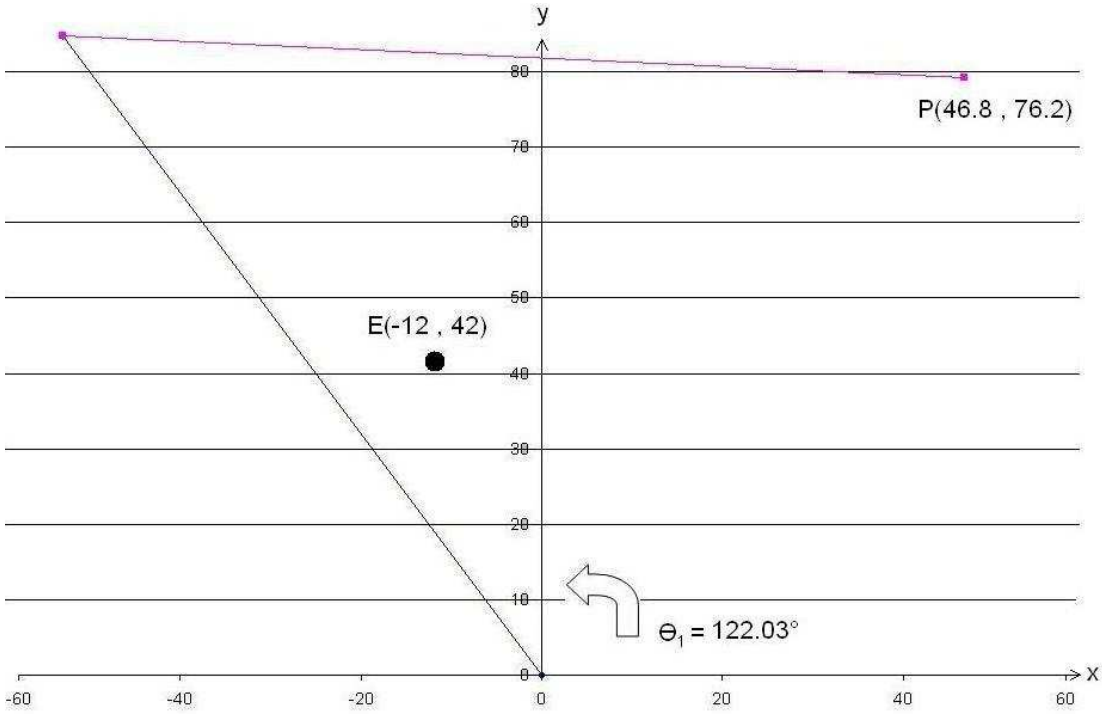

**Şekil 7.1 Robotun hareketinin ve engel koordinatının iki boyutlu grafiği**

Şekil 7.1' de, robot kolun saat yönünün tersine (pozitif yönde) döndüğünde, engel ile çarpışmadığı görülmektedir.

Hedef koordinatları: Px = -92.9 , Py = 76.5

Engel koordinatları: Ex = -98 , Ey = -67

Kolun hesaplanan $\theta_1$ açı değeri 104.48°' dir.




**Şekil 7.2 Robotun hareketinin ve engel koordinatının iki boyutlu grafiği**

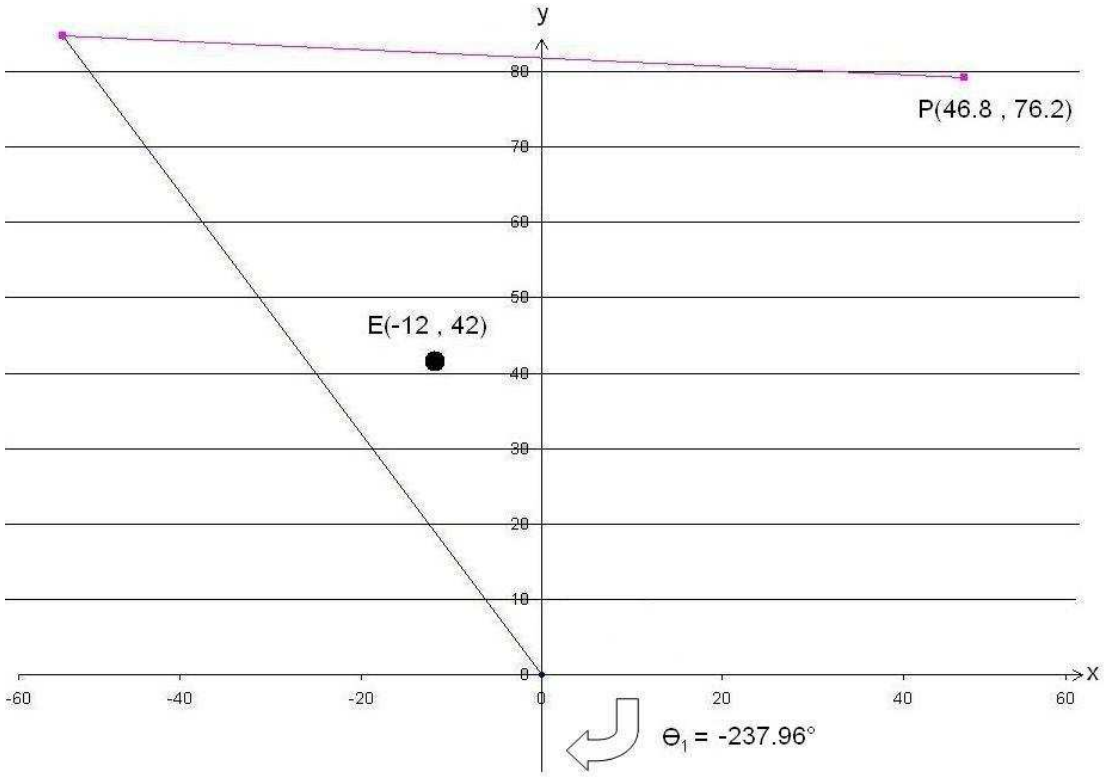
Şekil 7.2' de, robot kolun saat yönünün tersine (pozitif yönde) döndüğünde, engel ile çarpıştığı görülmektedir.

Hedef koordinatları: Px = -46.7 , Py = 76.2

Engel koordinatları: Ex = -12 , Ey = 42

Kolun hesaplanan $\theta_1$ açı değeri 122.03°' dir.

Bu açı değeri kullanıldığında çarpışma olduğundan, kolun saat yönünde (negatif yönde) döndürülmesi gerekmektedir. Çarpışma olmaması için $\theta_1$ ifadesinin alması gereken değeri -237.96º olmaktadır. Şekil 7.3' de görüldüğü gibi, robot kolun hareketi ok ile gösterildiği yönde olduğunda, hem birinci hem de ikinci kol engel ile çarpışmamaktadır.



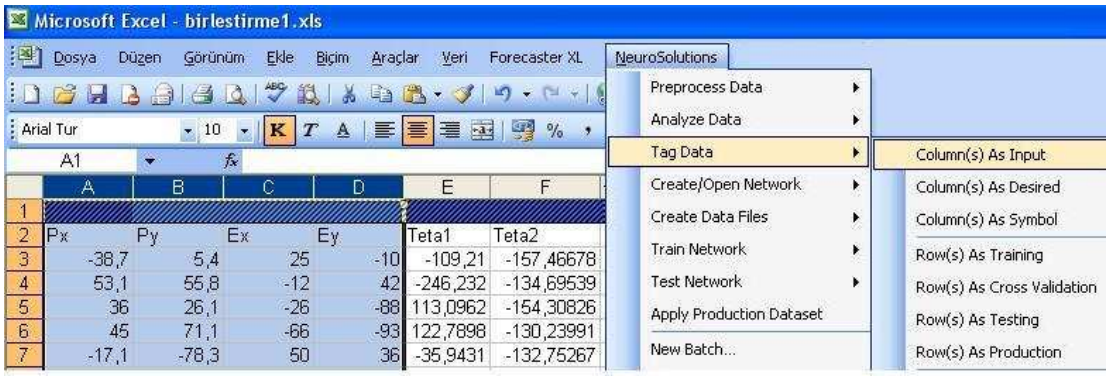

**Şekil 7.3 Çarpışmanın engellenmesi**

Yapay sinir ağları tarafından bulunması istenilen değerler, $\theta_1$ ve $\theta_2$ açı değerleridir. Bu değerlerin bulunması için ağa verilecek giriş değerleri ise, robotun hedef noktasının koordinatları (Px, Py) ve robotun çalışma alanı içerisinde bulunan engelin koordinatları (Ex, Ey)' dır.

Eğitim için NeuroDimension Inc. tarafından geliştirilen ve Windows işletim sistemi tabanlı olan NeuroSolutions V5.06 programı kullanılmıştır. Program kurulduktan sonra, programla birlikte gelen Excel modülü çalıştırılarak, modülün Excel programına menü olarak eklenmesi sağlanır.

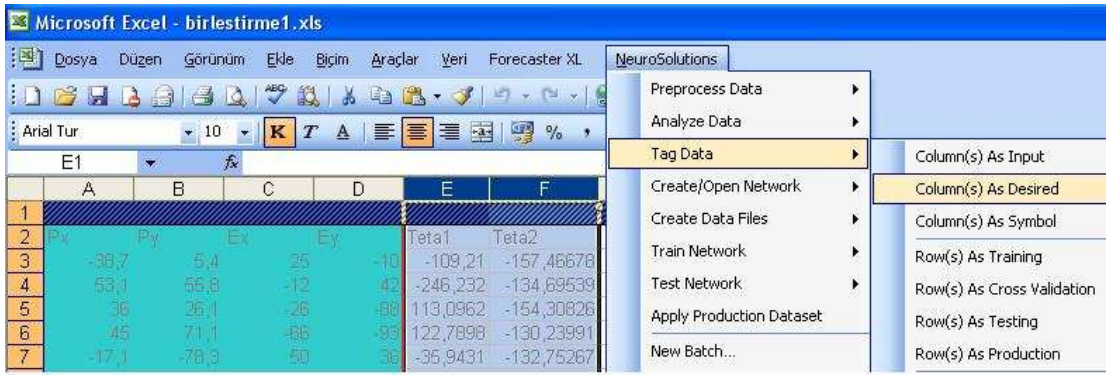
Excel' de oluşturulan 2000 adet örnek, ilgili hücreler alt alta gelecek şekilde sıralanır. 1950 adet örnek ağın eğitimi, 50 adet örnek de ağın testi için kullanılacaktır. NeuroSolutions programına, verilerin nasıl girildiği ve eğitimi gerçekleştirirken yapılan işlemler maddeler halinde açıklanmıştır:

1. Excel programı başlatılır ve hesaplanan değerlerin olduğu dosya açılır.
2. YSA için giriş değerleri olan, hedef ve engel koordinatlarının yazılı olduğu sütunlar seçilir. Menü çubuğuna yerleşmiş olan NeuroSolutions butonuna basılır. “Tag Data” üzerinde durularak “Coloumn(s) As Input” seçilir.



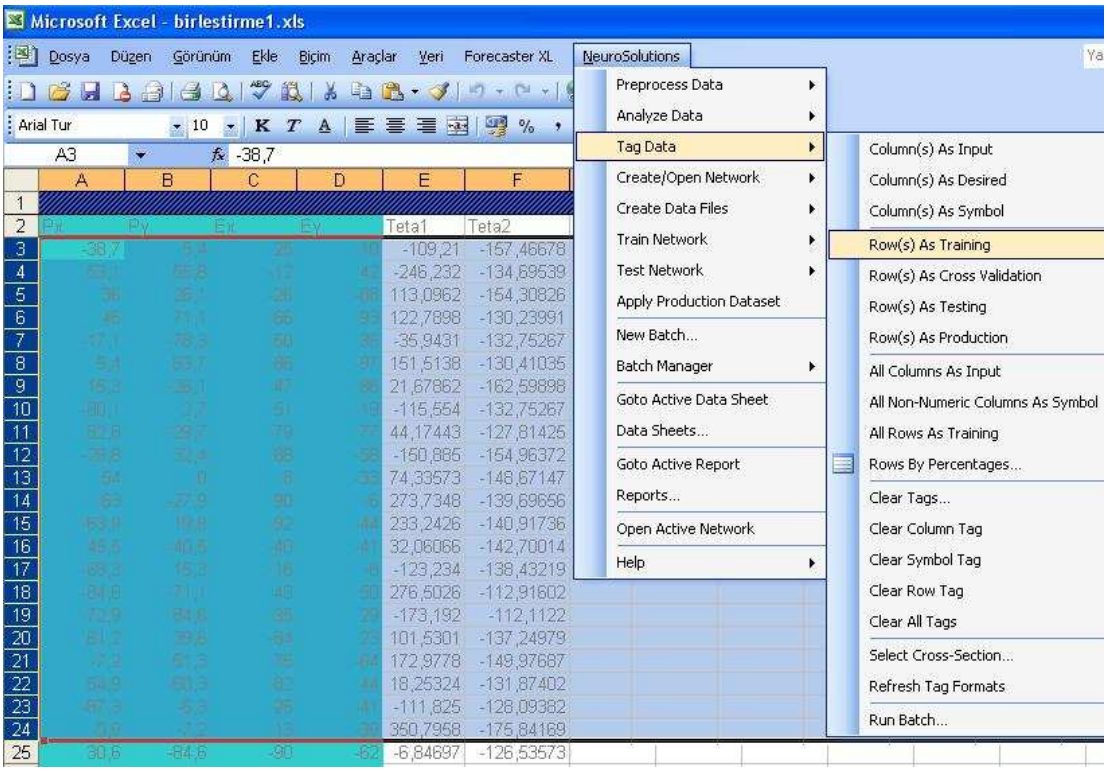

**Şekil 7.4 YSA giriş verilerinin seçilmesi**

3. YSA için çıkış değerleri olan, $\theta_1$ ve $\theta_2$ açı değerlerinin yazılı olduğu sütunlar seçilir. Menü çubuğuna yerleşmiş olan NeuroSolutions butonuna basılır. “Tag Data” üzerinde durularak “Coloumn(s) As Desired” seçilir.



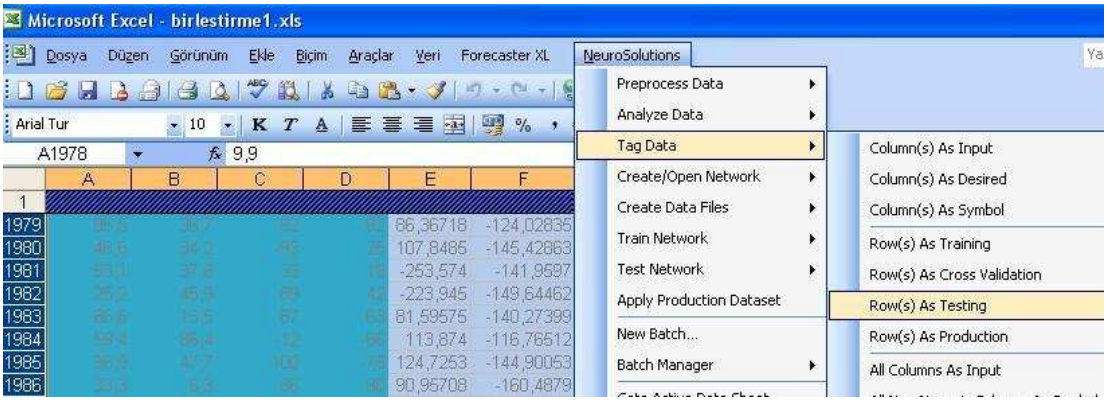

**Şekil 7.5 YSA çıkış verilerinin seçilmesi**

4. YSA için eğitim setini oluşturacak satırlar seçilir. “Tag Data” üzerinde durularak “Row(s) As Training” seçilir.



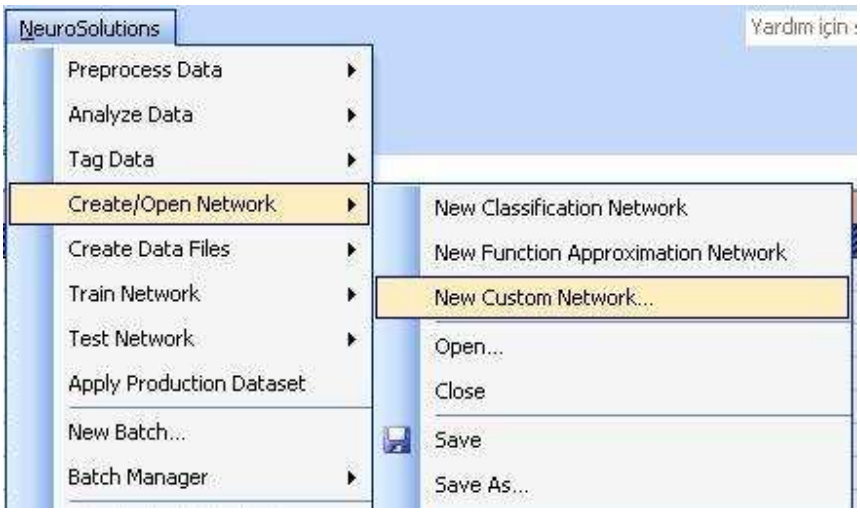

**Şekil 7.6 YSA eğitim verilerinin seçilmesi**

5. YSA için test setini oluşturacak satırlar seçilir. “Tag Data” üzerinde durularak “Row(s) As Testing” seçilir.



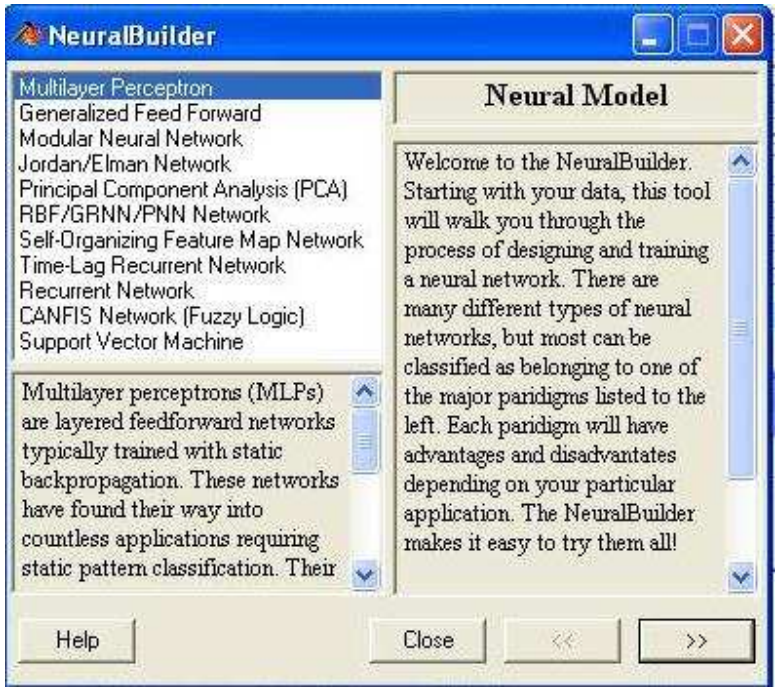

**Şekil 7.7 YSA test verilerinin seçilmesi**

6. “Create / Open Network” altında bulunan “New Custon Network” seçilir.



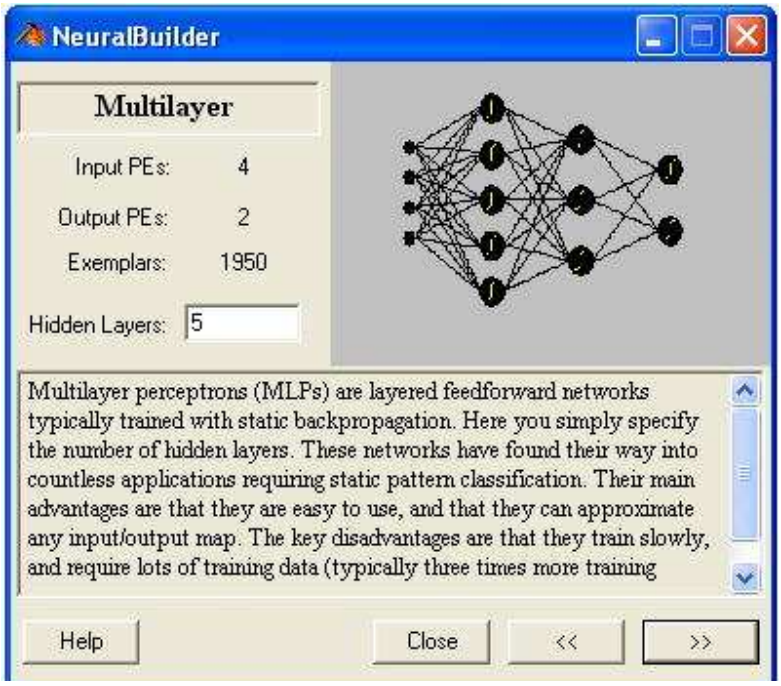

**Şekil 7.8 Ağın oluşturulması**

7. Açılan pencerede öğrenme modeli olarak “Multilayer Perceptron” seçilir ve ileri butonuna basılır.



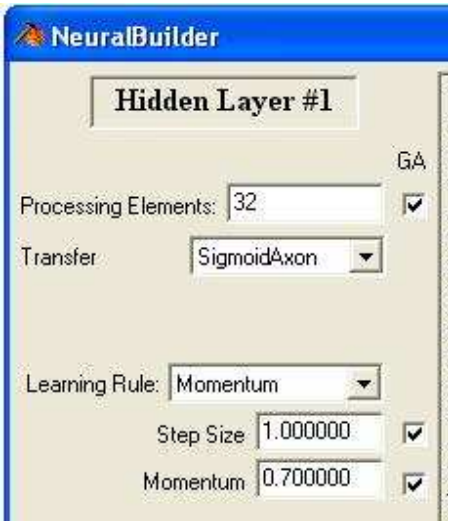

**Şekil 7.9 Model seçimi**

8. Açılan pencere öğrenme algoritmasının giriş ve çıkış eleman sayıları ve örnek sayısının gösterildiği, ayrıca gizli katman sayısının seçiminin yapıldığı yerdir. Burada gizli katman eleman sayısı 5 seçilmiştir.

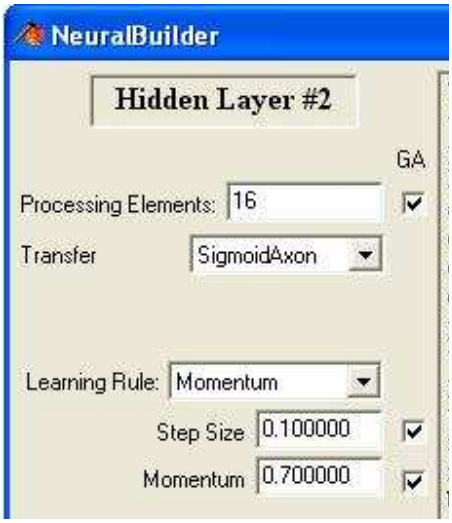



**Şekil 7.10 Gizli katman sayısı seçimi ve ağ özellikleri**

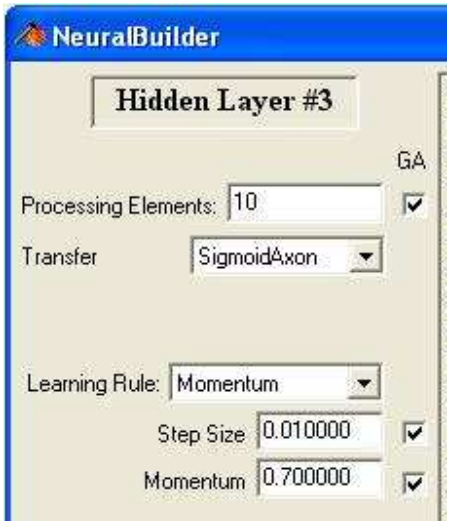
9. İleri butonuna basarak 1. gizli katman ayar penceresi açılır.



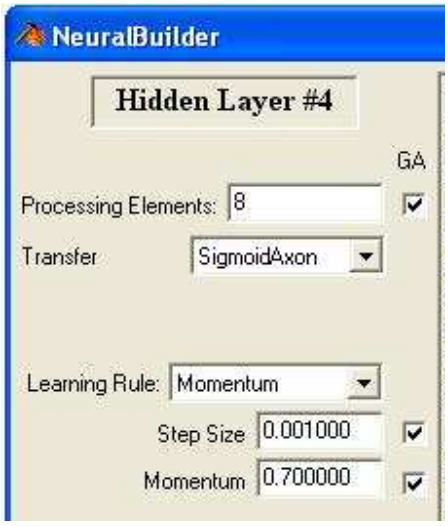

**(a)**



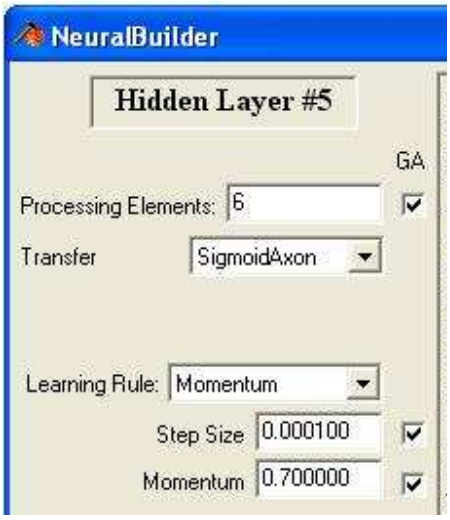

**(b)**




**(c)**




**(ç)**

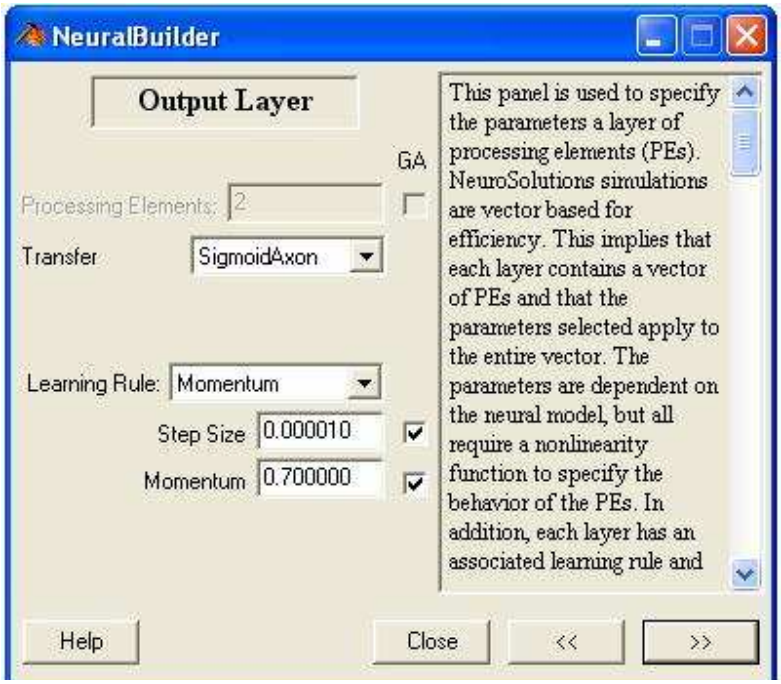



**(d)**

**Şekil 7.11 Gizli katman ayar pencereleri (a) 1. gizli katman, (b) 2. gizli katman, (c) 3. gizli katman, (ç) 4. gizli katman, (d) 5. gizli katman**

1. Gizli katmanın ayarı yapıldıktan sonra ileri butonuna basılarak bir sonraki gizli katman ayar penceresine geçilir. Böylece 5 adet gizli katmanın parametreleri, birbirinden bağımsız bir şekilde programa girilir. GA kutucuğu, YSA hesaplamaları yapılırken, ilgili parametrelerin genetik algoritma ile optimizasyona tabi tutulması için işaretlenir.

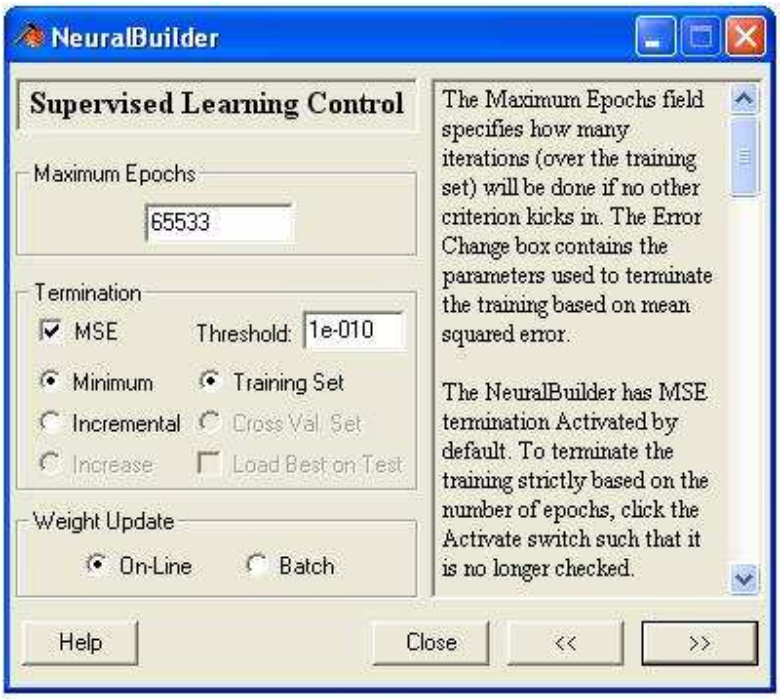
10. Output Layer penceresinde, çıkış katmanı parametrelerinin ayarları yapılır.




**Şekil 7.12 Çıkış katmanı parametreleri**

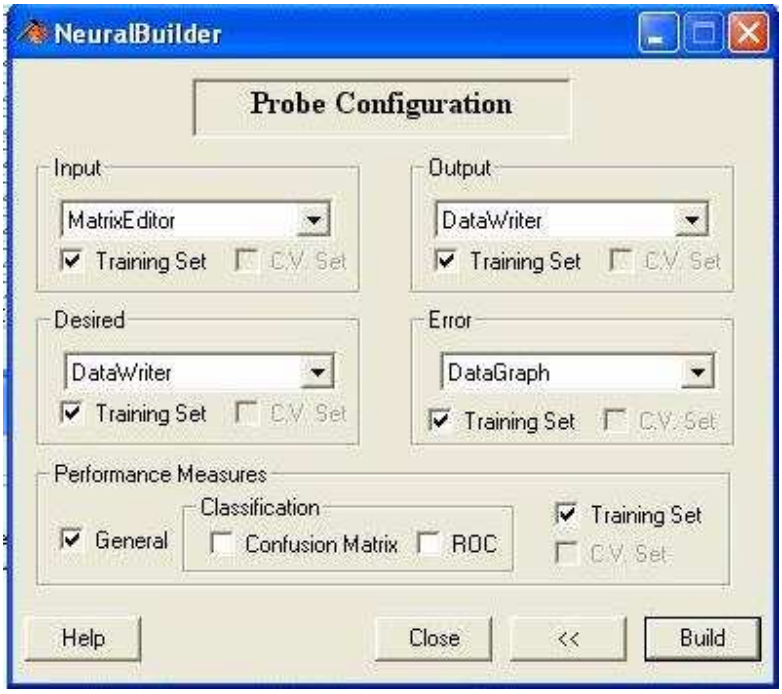
11. Supervised Learning Control (danışmanlı öğrenme) penceresinde iterasyon sayısı, programın desteklediği maksimum sayı olan 65533 olarak belirlenmiştir. Öğrenme kuralı MSE değeri 0.0000000001 olarak seçilmiştir. Durdurma kriteri olan "minimum", hesaplanan hatanın verilen eşik değerden düşük olduğu durumda, "incremental" ise o anki iterasyon ile bir önceki iterasyonun hatalarının farkı, verilen eşik değerden düşük olduğu durumda eğitimi durdurur. Ağırlık güncelleme seçeneklerinden "On-Line", her iterayonun sonunda, "Batch" ise belli sayıda iterasyondan sonra ağırlıkları günceller.

Yapılan hesaplamalarda durdurma kriteri "minimum", ağırlık güncelleme kriteri "On-Line" seçilmiştir.




**Şekil 7.13 Danışmanlı öğrenme kontrolü parametreleri**

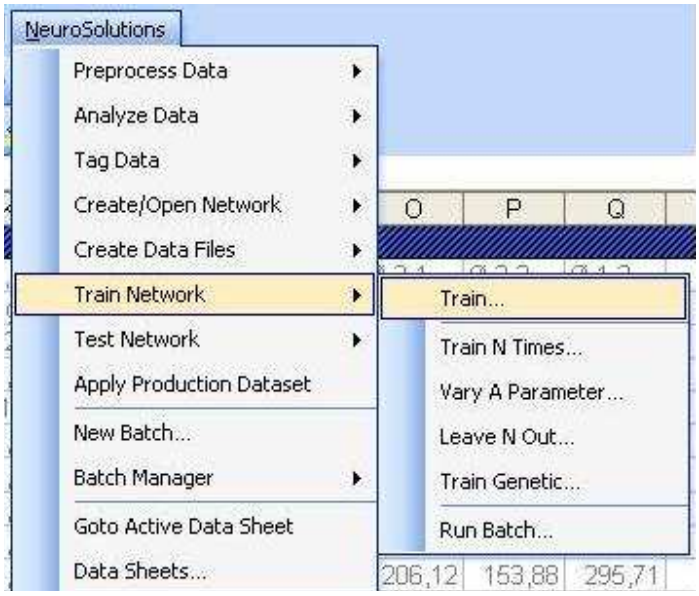
12. Probe Configuration penceresinde, eğitim esnasında meydana gelen değişiklikleri gösteren pencereler seçilir.




**Şekil 7.14 İnceleme pencereleri seçimi**

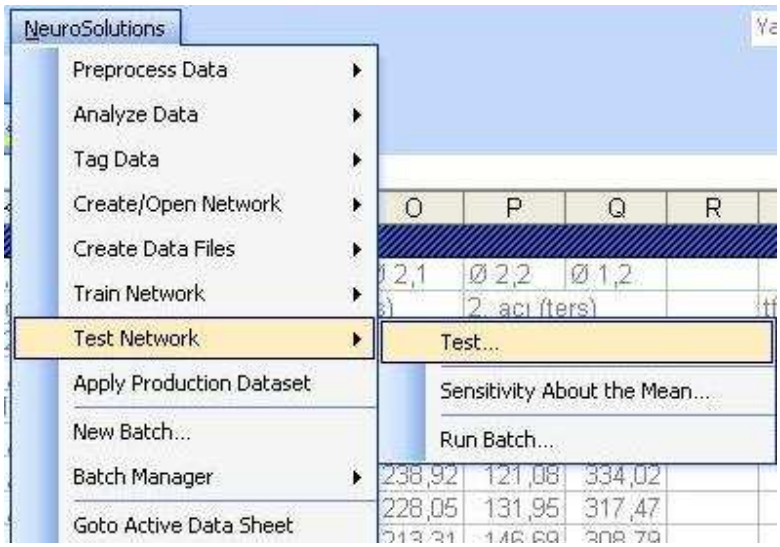
Değişiklikleri göstermek için kullanılacak pencereler işaretlendikten sonra Build butonuna basılarak ağ oluşturulur. Programın oluşturduğu ağ EK-4'de gösterilmiştir. Ayrıca ağın şematik gösterimi EK-5' de verilmiştir.

13. Ağ oluşturulduktan sonra eğitime başlanır. Bunun için "Train Network" altında bulunan "Train" seçilir ve eğitim başlatılır. Bazı değerlerin kaydedilmesini onayladıktan sonra eğitim başlar. Eğitim bittiğinde program, verilerin bulunduğu Excel dosyasında "Train Report" sekmesi açarak eğitimin hata grafiğini oluşturur. Ayrıca "Train MSE" sekmesi oluşturarak içine, her iterasyon sonucu elde edilen hata miktarını yazar.



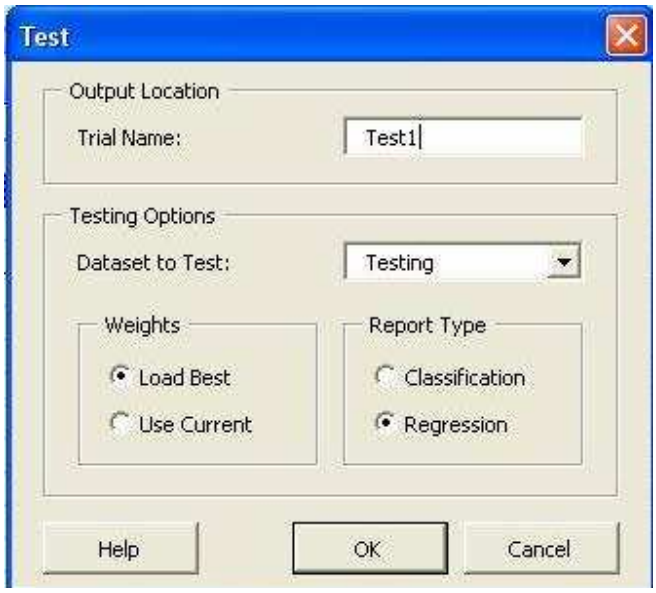

**Şekil 7.15 Eğitimin başlatılması**

14. Eğitim işlemi bittikten sonra, ağı test etmek için “Test Network” altında bulunan “Test…” seçilir.




**Şekil 7.16 Testin başlatılması**

Test seçenekleri penceresinde “Load Best” ve “Regression” seçilir. Tamam butonuna basıldığında program otomatik olarak test sonuçlarını oluşturur ve verilerin bulunduğu Excel dosyasına “Test Report” sekmesini ekleyerek, eğitim ve test sonuçları karşılaştırma grafiğini bu sekmenin içine yazar. Ayrıca “Test IOData” sekmesini oluşturarak, istenilen ve testten sonra elde edilen sonuç değerlerini içine yazar.




**Şekil 7.17 Test seçeneklerinin seçimi**

# 8. SONUÇ ve ÖNERİLER

NeuroSolutions kullanılarak YSA hesaplamalarının yapıldığı bilgisayarın donanım özellikleri aşağıda verilmiştir:

- Intel Pentium(R) 4 CPU 2.80 GHz Prescott İşlemci
- 512 Mb DDR Ram
- GForce FX 5200 128 Mb Ekran Kartı

Yapılan ilk eğitimde oluşturulan ağın özellikleri, aşağıda ve Tablo 8.1' de gösterilmiştir:

Maksimum iterasyon sayısı: 65533
Hata fonksiyonu: MSE
İstenilen minimum hata: 0.00001

**Tablo 8.1 Ağ için belirlenen parametreler**

| | **Giriş Katmanı** | **1. Gizli katman** | **Çıkış Katmanı** |
|---|---|---|---|
| Eleman Sayısı | 4* | 15* | 2 |
| Transfer Fonksiyonu | | SigmoidAxon | SigmoidAxon |
| Öğrenme Kuralı | | momentum | Momentum |
| Öğrenme Katsayısı | | 0,7* | 0,7* |
| Adım | | 1* | 0,1* |

(*) ile işaretli değerler, eğitim esnasında programın bir özelliği olarak, genetik algoritma ile optimizasyona tabi tutulmaktadır. Eğitim, yaklaşık olarak 22 dakika sürmüş ve eğitim sonucunda elde edilen en küçük hatanın 0.0088 olduğu görülmüştür. Test işleminden sonra ise hatanın çok büyük olduğu görülüp, eğitim işlemine parametreler değiştirerek devam edilmiştir.

Yapılan 50 farklı deneme eğitimi sonucunda en küçük hatanın elde edildiği ağın özellikleri, aşağıda ve Tablo 8.2' de verilmiştir:

Maksimum iterasyon sayısı: 65533

Hata fonksiyonu: MSE

İstenilen minimum hata: 0.0000000001

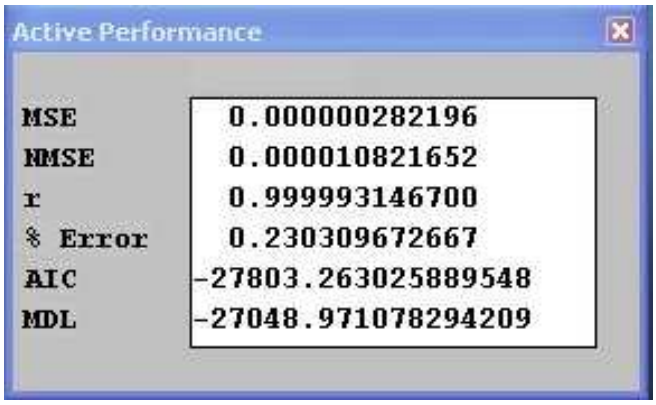
**Tablo 8.2 Ağ için belirlenen parametreler**

| | **Giriş Katmanı** | **Gizli Katmanlar** | | | | | **Çıkış Katmanı** |
|---|---|---|---|---|---|---|---|
| | | **1** | **2** | **3** | **4** | **5** | |
| Eleman Sayısı | 4* | 32* | 16* | 10* | 8* | 6* | 2 |
| Transfer Fonksiyonu | | SigmoidAxon | SigmoidAxon | SigmoidAxon | SigmoidAxon | SigmoidAxon | SigmoidAxon |
| Öğrenme Kuralı | | Momentum | Momentum | Momentum | Momentum | Momentum | Momentum |
| Öğrenme Katsayısı | | 0.7* | 0.7* | 0.7* | 0.7* | 0.7* | 0.7* |
| Adım | | 1 | 0.1 | 0.01 | 0.001 | 0.0001* | 0.00001* |

Deneme eğitimlerinin özellikleri ve elde edilen hata değerleri EK-6' da sunulmuştur.

Eğitim, yaklaşık olarak 4 saat 25 dakika sürmüştür. Şekil 8.1' de görüldüğü gibi yapay sinir ağı modelinin 0.00000028 hata ile sonuca ulaştığı görülmüştür. Hata oranı % 0.23' tür.

Ağın eğitimi için gerekli olan giriş değerleri Px, Py, Ex ve Ey ile istenilen çıkış değerleri $\theta_1$ ve $\theta_2$ tablo halinde EK-1' de verilmiştir. Örnek sayısı çok olduğundan ilk 230 örnek sunulmuştur.

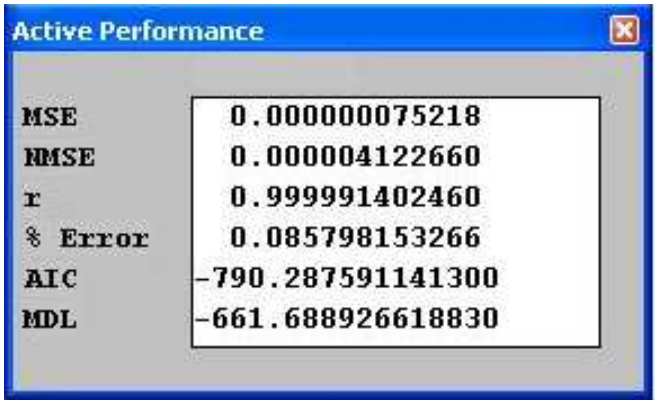



**Şekil 8.1 Eğitim performans tablosu**

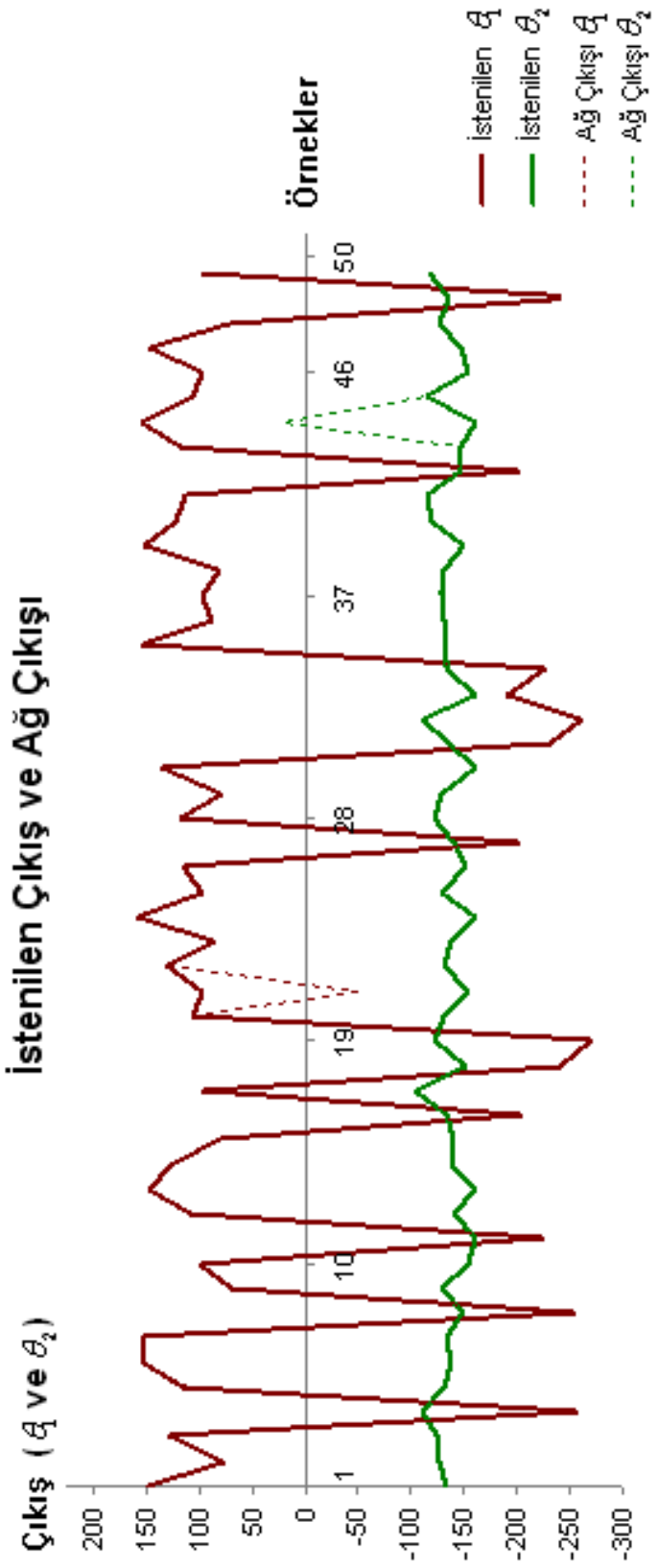

**Şekil 8.2 Eğitim sonucu iterasyon – aktif değer grafiği**

Şekil 8.2, eğitim sonucu elde edilen aktif değer' in iterasyona göre değişim grafiğidir. Burada da görüldüğü gibi, eğitim yaklaşık olarak 100. iterasyondan sonra 0 (sıfır) hata mertebesinde devam etmiştir.

Ağın eğitimi tamamlandıktan sonra yapılan test işleminin sonucunda ağın, verilen test örneklerini çok düşük bir hatayla öğrendiği görülmüştür. Test sonucu elde edilen performans tablosu, Şekil 8.3' de gösterilmiştir.




**Şekil 8.3 Test performans tablosu**




**Şekil 8.4 İstenilen ve YSA ile hesaplanan değerler karşılaştırma grafiği**

Şekil 8.4’ de gösterilen grafikte, YSA’ ya verilen ve YSA’ dan alınan $\theta_1$ ve $\theta_2$ açı değerleri karşılaştırılmıştır. Kırmızı çizgi ile istenilen $\theta_1$ ve yeşil çizgi ile istenilen $\theta_2$ ifade edilmektedir. Kesikli çizgilerle gösterilen YSA çıkışlarının, sürekli çizgilerle gösterilen istenilen değerlerle çakışık olması, eğitimin çok düşük bir hatayla yapıldığını göstermektedir.

Yapay sinir ağlarının, veriler arasında doğrudan bir ilişki bulunmadığı durumlarda, doğruya çok yakın sonuçlar verdiği görülmüştür.

Ancak Tablo 8.3’ de görüldüğü gibi ağın her zaman doğru sonuç vermesi beklenemez. Px, Py, Ex ve Ey giriş değerleri için, hesaplanan $\theta_1$ açı değeri -49.616º’ dir. Ancak YSA tarafından bulunan $\theta_1$ açı değeri 97.533°’ dir. Aynı şekilde hesaplanan $\theta_2$ açı değeri 21.418° olmasına rağmen, YSA’ dan elde edilen sonuç -160.641°’ dir.

Elde edilen bu verilere dayanılarak, YSA’ nın çıkış değerlerine, özellikle insan hayatını ilgilendiren önemli konularda doğrudan güvenmenin uygun olmayacağı söylenebilir. Ağın çıkış değerini doğrudan sonuç olarak değil, sonuç için yol gösterici bir değer olarak kabul etmek daha uygun olacaktır.

**Tablo 8.3 Ağ çıkışı ve istenilen değer karşılaştırma tablosu**

| **Px** | **Py** | **Ex** | **Ey** | **Y** $\theta_1$ | **Y** $\theta_2$ | **İ** $\theta_1$ | **İ** $\theta_2$ | **H** $\theta_1$ | **H** $\theta_2$ |
|---|---|---|---|---|---|---|---|---|---|
| 41.4 | 15.3 | -45 | -20 | 97.533 | -154.501 | -49.616 | -154.5 | -2.96577 | 4.100E-05 |
| 9 | 32.4 | 5 | -56 | 154.796 | -160.641 | 154.797 | 21.418 | -4.89E-06 | -8.500190 |

**Y** $\theta_1$ **,** **Y** $\theta_2$ **:** YSA ile elde edilen açı değerleri

**İ** $\theta_1$ **,** **İ** $\theta_2$ **:** Matematiksel hesaplama ile elde edilen ve YSA’ dan istenilen açı değerleri

**H** $\theta_1$ **,** **H** $\theta_2$ : İstenilen ve YSA ile elde edilen değerler arasındaki hata oranı

PID kontrolörde, parametrelerin dengeli bir şekilde ayarlanmasıyla uygun bir kontrolör elde edilebilir. Eğer bu katsayılar dengeli bir şekilde ayarlanmazsa, PID kontrolörün sağlayacağı üstün özelliklerden yararlanılamaz.

PID tipi denetleyiciler, denetlenen proses kararlı bir durumdayken işini iyi yaparken aşağıdaki durumlarda işin üstesinden gelemezler:

- Güçlü dış etkinin olduğu durumlarda (doğrusal olmayan durum),
- Süreçte zamana göre değişen parametreler varsa,
- Ölü zamanların bulunması.

Bunun sebebi, PID denetleyicinin, işlemin, her durumda kesin bir doğrusallık özelliği göstereceğini varsaymasıdır. Bu kabul kararlı durumlarda geçerli olurken, güçlü bozucu faktörler, işlem noktasını orijininden saptırabilir. İşte bu anda doğrusal kabul daha fazla geçerliliğini koruyamamaktadır. Aynı olay işlem parametrelerinin zamanla değişmesiyle de gerçekleşir.

Ancak YSA uygulamalarında, uygulama esnasında doğrusallık gerekmez. Problemin çözüm kümesinde meydana gelebilecek bozulmalar ağ tarafından tolere edilebilir, eksik veya hatalı girilen bilgiler düzeltilir ya da tamamlanabilir. Sistem matematiksel olarak modellenmek zorunda olmadığından, matematiksel işlemler yapılırken yapılan ihmaller, YSA uygulamalarında ihmal edilmeyerek hesaplamaya dâhil olmaktadır. Böylece sistem çıkış değerinin daha düşük hata ile bulunması sağlanır.

Bu çalışmada elde edilen sonuçlara yakın değerler, PID gibi matematiksel tabanlı veya Bulanık Mantık (Fuzzy Logic) gibi yapay zeka tabanlı kontrol yöntemleri kullanılarak elde edilebilir. Elde edilen yeni değerler, bu çalışmada bulunan değerlerle karşılaştırılarak çözüm yöntemlerinin birbirlerine karşı üstünlükleri değerlendirilebilir.

# 9. KAYNAKLAR

[1] Aydın, S. ve Temeltaş, H., Mobil robotlarda diferansiyel evrim ile optimal yörünge planlama, İtü Dergisi/d, Mühendislik, Cilt. 3, Sayı. 1, Sayfa 43-54, 2004.

[2] Bingül, Z. ve Küçük, S., Robot Tekniği I, Birsen Yayınevi, 2005.

[3] Blackmore, L. and Williams, B., Optimal manipulator path planning with obstacles using disjunctive programming, IEEE, 2006.

[4] Bolat, B., Erol, K. O. ve İmrak, C. E., Genetic algorithms in engineering applications and the function of operators, Sigma Journal of Engineering and Natural Sciences, 2004.

[5] Cherif, A. R., Perdereau, V. and Drouin, M., Inverse kinematic at acceleration level using neural network, Lab. PARC, University P. & M. Curie, France, 1995.

[6] Çonkur, E. Ş., Gereğinden çok serbestlik dereceli robot kollarının yörünge planlaması için geliştirilmiş bir yazılım, 11. Ulusal Makine Teorisi Sempozyumu, Gazi Üniversitesi Mühendislik Mimarlık Fakültesi, 2003.

[7] Duran, M. A., Puma tipi bir manipülatörün kontrolü, Yüksek Lisans Tezi, Selçuk Üniversitesi Fen Bilimleri Enstitüsü, 2007.

[8] Er, M. J. and Liew, K. C., Control of adept one SCARA robot using neural networks, IEEE, 0278-0046/97, 1997

[9] Fu, K. S., Gonzalez, R.C. and Lee, C. S. G., Robotics: Control, Sensing, Vision and Intelligence, McGraw-Hill Book Company, New York, 1987.

[10] Galicki, M., Collision-free control of robotic manipulators in the task space, Journal of Robotic Systems, 22(8), pp. 439-455, 2005.

[11] Goldberg, D. E., Genetic Algorithms in Searh Optimization and Machine Learning, Addison – Wesley, 1989.

[12] Hacıoğlu, Y., Bir robotun bulanık mantıklı kayan kipli kontrolü, Yüksek Lisans Tezi, İstanbul Üniversitesi Fen Bilimleri Enstitüsü, 2004.

[13] Haupt, R. L. and Haupt, S. E., Practical Genetic Algorithms, John Wiley & Sons Inc, 1998.

[14] Kert, M., Gerçek görüntüden elde edilen koordinatlarla robot kol hareket optimizasyonu, Yüksek Lisans Tezi, Mustafa Kemal Üniversitesi Fen Bilimleri Enstitüsü, 2006.

[15] Khoogar, A. R. and Parker, J. K., Obstacle avoidance of redundant manipulators using genetic algorithms, IEEE, CH2998-3/91/0000-0317, 1991.

[16] Laribi, M.A., Romdhane, L. and Zeghloul, S., Analysis and dimensional synthesis of the DELTA robot for a prescribed workspace, Mechanism and Machine Theory, Vol. 42, pp. 859–870, 2007.

[17] Lee, S. and Kardaras, G., Collision-free path planning with neural networks, International Conference of Robotics and Automation Albuquerque, New Mexico, 1997.

[18] Lynch, K. M., Shiroma, N., Arai, H. and Tanie, K., Collision-free trajectory planning for a three-dof robot with a passive joint, International Journal of Robotics Research, 2000.

[19] Mitchell, T. M., Machine Learning, McGraw-Hill Book Company, New York, 1997.

[20] Nabiyev, V. V., Yapay Zeka, Seçkin Yayıncılık, Ankara, 2003.

[21] Nearchou, A. C., Solving the inverse kinematics problem of redundant robots operating in complex environments via a modified genetic algorithm, Mechanism and Machine Theory, Vol. 33, No. 3, pp. 273–292, 1998.

[22] Noguchi N., Terao, H., Path planning of an agricultural mobile robot by neural network and genetic algorithm, Computers and Electronics in Agriculture, No. 18, pp. 187–204, 1997.

[23] Öztemel, E., Yapay Sinir Ağları, Papatya Yayınları, 2003.

[24] Öztürk, S., Scara tipi robotun yapay sinir ağları ile eğitilmesi, Yüksek Lisans Tezi, Sakarya Üniversitesi Fen Bilimleri Enstitüsü, 2007.

[25] Pashkevich, A., Kazheunikau, M. and Ruano, A. E., Neural network approach to collision free path-planning for robotic manipulators, International Journal of Systems Science, Vol. 37, No. 8, pp. 555–564, 2006.

[26] Risse, W. and Hiller, M., Dextrous motion control of redundant SCARA robot, IEEE, 0-7803-4503-7/98, 1998.

[27] Sağıroğlu, Ş., Beşdok, E. ve Erler, M., Mühendislikte Yapay Zeka Uygulamaları I, Yapay Sinir Ağları, Ufuk Kitap Kırtasiye Yayıncılık Tic. Ltd., 2003.

[28] Shibata, T., Abe, T., Tanie, K. and Nose, M., Motion planning by genetic algorithm for a redundant manipulator using a model of criteria of skilled operators, Information Sciences, pp. 102, 171–186, 1997.

[29] Şahin, Y., Scara tipi bir robotun yörünge kontrolünde PID kontrol uygulaması, Yüksek Lisans Tezi, Selçuk Üniversitesi Fen Bilimleri Enstitüsü, 2006.

[30] Şen, Z., Genetik Algoritmalar ve En İyileme Yöntemleri, Su Vakfı Yayınları, İstanbul, 2004.

[31] Tiryaki, A. ve Kazan, R., Scara robot dinamiğinin YSA kullanılarak modellenmesi Mühendis ve Makina, Cilt. 46, Sayı. 550, 2005.

[32] Toogood, R., Hao, H. and Wong, C., Robot path planning using genetic algorithms, IEEE, 0-7873-2559-1/95, 1995.

[33] Visioli, A. and Legnani, G., Trajectory tracking control of industrial SCARA robot manipulators, IEEE; 0278-0046/02, 2002.

[34] Zalzala, A. M. S. and Morris, A. S., Neural Networks for Robotic Control, Theory and Applications, Elli Horwood, 1996.

[35] Zhang, H., Liu, M., Liu, R. and Hu, T., Path planning of robot in three-dimensional grid environment based on genetic algorithms, IEEE, 978-1-4244-2114-5/08, 2008.

[36] http://www.mmo.org.tr/muhendismakina/arsiv/2001/ekim/Genetik_Algoritma.htm (2008).

[37] http://www.nd.com (2008).

# 10. EKLER

## EK-1 Yapay Sinir Ağları Eğitim Seti

| | Giriş Değerleri | | | | İstenilen Açı Değerleri | |
|---|---|---|---|---|---|---|
| | Hedef Koordinatları (mm) | | Engel Koordinatları (mm) | | $\theta_1$ (°) | $\theta_2$ (°) |
| **Sıra No** | **Px** | **Py** | **Ex** | **Ey** | **İ $\theta_1$** | **İ $\theta_2$** |
| *1* | -38.7 | 5.4 | 25 | -10 | -109.2100817 | -157.4667803 |
| *2* | 53.1 | 55.8 | -12 | 42 | -246.2320399 | -134.6953892 |
| *3* | 36 | 26.1 | -26 | -88 | 113.0962399 | -154.308256 |
| *4* | 45 | 71.1 | -66 | -93 | 122.789767 | -130.2399054 |
| *5* | -17.1 | -78.3 | 50 | 36 | -35.94310887 | -132.7526728 |
| *6* | 5.4 | 83.7 | 86 | -97 | 151.5137907 | -130.4103533 |
| *7* | 15.3 | -26.1 | 47 | 86 | 21.67861556 | -162.5989791 |
| *8* | -80.1 | 2.7 | 51 | -19 | -115.5542511 | -132.7526728 |
| *9* | 82.8 | -29.7 | -79 | 77 | 44.17443355 | -127.8142461 |
| *10* | -28.8 | 32.4 | 88 | -58 | -150.8846007 | -154.9637198 |
| *11* | 54 | 0 | 8 | -33 | 74.33573315 | -148.6714663 |
| *12* | -63 | -27.9 | 90 | -6 | 273.7347801 | -139.6965604 |
| *13* | -63.9 | 19.8 | -92 | -44 | 233.2426083 | -140.9173637 |
| *14* | 49.5 | -40.5 | -40 | -41 | 32.06066495 | -142.7001436 |
| *15* | -69.3 | 15.3 | -16 | -8 | -123.2339007 | -138.4321917 |
| *16* | -84.6 | -71.1 | 43 | 50 | 276.5025611 | -112.91602 |
| *17* | -72.9 | 84.6 | -35 | 29 | -173.1923558 | -112.1121975 |
| *18* | 61.2 | 39.6 | -84 | 23 | 101.530137 | -137.2497882 |
| *19* | -7.2 | 51.3 | 75 | -64 | 172.9777626 | -149.9768716 |
| *20* | 54.9 | -60.3 | -82 | 44 | 18.25323527 | -131.8740209 |
| *21* | -87.3 | -6.3 | 25 | -41 | -111.825498 | -128.0938206 |
| *22* | -0.9 | -7.2 | 13 | -39 | 350.7958273 | -175.8416873 |
| *23* | 30.6 | -84.6 | -90 | -62 | -6.8469685 | -126.5357328 |
| *24* | -25.2 | -28.8 | -71 | 59 | -52.21707377 | -157.9377028 |
| *25* | -68.4 | 65.7 | -78 | -11 | -162.1545525 | -123.3839942 |
| *26* | 36.9 | 10.8 | -98 | 82 | -264.7696895 | -157.8329162 |
| *27* | -36 | -83.7 | 90 | -89 | 309.6256063 | -125.7970207 |
| *28* | -64.8 | -9.9 | 75 | -81 | 259.5535157 | -141.7343223 |
| *29* | -21.6 | -1.8 | -28 | -69 | -91.45797082 | -167.556775 |
| *30* | -52.2 | -50.4 | 82 | -42 | 292.7225622 | -137.4552964 |
| *31* | 88.2 | 51.3 | -57 | 17 | 89.50853832 | -118.6497153 |
| *32* | -69.3 | 11.7 | 72 | -60 | -120.1561352 | -138.853619 |
| *33* | -12.6 | 46.8 | 58 | 55 | -178.9557543 | -151.9515151 |
| *34* | 33.3 | 10.8 | 14 | 47 | -262.1117212 | -159.8382781 |
| *35* | 27.9 | 3.6 | -37 | -28 | 89.26654272 | -163.8283267 |
| *36* | -47.7 | -45.9 | 53 | 32 | -65.43040914 | -141.342594 |
| *37* | -22.5 | -71.1 | 75 | -47 | 320.5466315 | -136.2138072 |
| *38* | 62.1 | 15.3 | -22 | 11 | 85.19076731 | -142.7001436 |
| *39* | 20.7 | 11.7 | 84 | 65 | -247.3520761 | -166.3440699 |
| *40* | -69.3 | -81 | 9 | -91 | -72.75709257 | -115.5834669 |

| Sıra No | Px | Py | Ex | Ey | İ $\theta_1$ | İ $\theta_2$ |
|---|---|---|---|---|---|---|
| *41* | 67.5 | -56.7 | -63 | -1 | 23.816698 | -127.6939146 |
| *42* | -55.8 | 17.1 | -97 | -95 | 235.9960704 | -146.0676765 |
| *43* | -55.8 | 23.4 | -94 | 49 | 229.6388431 | -144.7796388 |
| *44* | 13.5 | 54.9 | 88 | -89 | 149.7649105 | -147.1598716 |
| *45* | 44.1 | 71.1 | 89 | 61 | -236.538626 | -130.5415265 |
| *46* | 25.2 | -15.3 | -45 | 60 | 50.25969284 | -163.0468491 |
| *47* | 1.8 | 40.5 | 92 | -58 | 165.7603091 | -156.610227 |
| *48* | -61.2 | -33.3 | 52 | 79 | -81.83595864 | -139.225299 |
| *49* | 39.6 | -40.5 | 1 | 46 | 27.90412552 | -147.0957425 |
| *50* | -55.8 | 0.9 | 67 | 64 | -107.1267411 | -147.5946084 |
| *51* | -6.3 | -54.9 | 10 | -45 | -22.58587883 | -147.9208239 |
| *52* | 36.9 | -47.7 | 36 | -94 | 20.17525823 | -144.9005264 |
| *53* | 40.5 | 46.8 | 50 | 95 | -238.8987833 | -143.9472499 |
| *54* | 26.1 | -66.6 | -52 | 57 | 0.443374432 | -138.0871286 |
| *55* | 28.8 | -32.4 | 46 | 49 | 29.11539926 | -154.9637198 |
| *56* | -65.7 | -73.8 | 54 | -80 | 288.7168052 | -120.7873695 |
| *57* | 9.9 | -67.5 | -97 | 91 | -11.60066592 | -140.110885 |
| *58* | -35.1 | -22.5 | 92 | 44 | 290.6286627 | -155.9355 |
| *59* | 18 | -8.1 | 7 | 36 | 60.10835909 | -168.6722088 |
| *60* | 34.2 | -76.5 | -31 | -49 | -0.682981688 | -130.459132 |
| *61* | 76.5 | 63.9 | -86 | 98 | 99.97867258 | -120.2137272 |
| *62* | 86.4 | -21.6 | -56 | -94 | 49.52151613 | -127.1155192 |
| *63* | 38.7 | -58.5 | -54 | 89 | 12.95533392 | -138.9383302 |
| *64* | 13.5 | 6.3 | -14 | -57 | 110.7450743 | -171.4563616 |
| *65* | -70.2 | 39.6 | -59 | -52 | 216.8069969 | -132.4689065 |
| *66* | -63.9 | 51.3 | 69 | -54 | 207.0545125 | -131.6251964 |
| *67* | 86.4 | -81.9 | -81 | -61 | 10.00164958 | -106.9400836 |
| *68* | -11.7 | 77.4 | 16 | -78 | 165.5544874 | -133.917152 |
| *69* | -0.9 | -17.1 | 75 | 91 | -7.924370224 | -170.1768346 |
| *70* | 0 | 18 | -98 | 97 | 174.8363929 | -169.6727858 |
| *71* | 3.6 | -27 | 8 | 76 | -0.233066684 | -164.3445799 |
| *72* | -6.3 | 79.2 | -8 | 47 | -198.8583865 | -133.1871422 |
| *73* | 81 | -57.6 | 64 | -95 | 24.78388257 | -120.4018757 |
| *74* | -5.4 | -24.3 | -33 | 70 | -19.67860258 | -165.7004103 |
| *75* | 26.1 | 2.7 | 41 | -20 | 88.36740694 | -164.9225316 |
| *76* | 58.5 | 63 | 92 | -22 | 111.6623003 | -129.0824079 |
| *77* | -54 | -66.6 | 10 | -6 | 295.5790072 | -129.2290402 |
| *78* | -85.5 | -66.6 | -3 | -29 | -84.89570831 | -114.3751554 |
| *79* | -55.8 | -45 | 62 | 68 | 287.8812075 | -137.9934221 |
| *80* | 41.4 | -65.7 | -45 | 45 | 9.369036501 | -134.3049809 |
| *81* | 36 | 67.5 | 87 | -17 | 129.4388884 | -135.0227507 |
| *82* | -68.4 | 77.4 | -23 | -16 | 190.372478 | -117.8095451 |
| *83* | -52.2 | -63.9 | 12 | -25 | -63.61084313 | -131.2691831 |
| *84* | 14.4 | -40.5 | 9 | 90 | 7.162340549 | -155.1784294 |
| *85* | -37.8 | -43.2 | 3 | 51 | -57.86519412 | -146.6414621 |
| *86* | -7.2 | -30.6 | -33 | -53 | -22.28366902 | -161.9137018 |
| *87* | 69.3 | -33.3 | 23 | 16 | -318.2733946 | -134.7835937 |
| *88* | 43.2 | 85.5 | 25 | 99 | -235.4239403 | -122.7636961 |
| *89* | -15.3 | 27 | -97 | 91 | 200.6122159 | -162.1468674 |

| Sıra No | Px | Py | Ex | Ey | İ $\theta_1$ | İ $\theta_2$ |
|---|---|---|---|---|---|---|
| *90* | -70.2 | -54.9 | -92 | 16 | -78.43391985 | -127.0776989 |
| *91* | -90 | -41.4 | 74 | -87 | 265.011104 | -120.6173475 |
| *92* | 64.8 | 83.7 | 62 | 76 | -249.7024091 | -116.0887925 |
| *93* | -68.4 | -88.2 | 42 | -87 | 288.2834489 | -112.1547839 |
| *94* | -1.8 | 18 | -94 | 72 | 180.5211618 | -169.6211372 |
| *95* | -80.1 | 81 | -55 | 81 | -170.0412649 | -110.5576398 |
| *96* | -67.5 | -65.7 | -82 | 77 | -73.87189705 | -123.8046462 |
| *97* | -9 | -25.2 | -62 | -90 | -27.34274837 | -164.6221514 |
| *98* | -64.8 | -19.8 | 31 | -93 | 267.1878014 | -140.3939562 |
| *99* | -26.1 | 13.5 | -54 | -100 | -125.7985536 | -163.1026444 |
| *100* | 13.5 | -45.9 | -56 | -46 | 2.548995654 | -152.3189106 |
| *101* | 42.3 | -88.2 | 73 | -46 | 356.340925 | -121.4378581 |
| *102* | -90 | 82.8 | 27 | -47 | -170.3099516 | -104.6082088 |
| *103* | 54.9 | -26.1 | -6 | 81 | 46.87862609 | -144.6110005 |
| *104* | 42.3 | 9 | -28 | -10 | 89.5235304 | -155.024104 |
| *105* | 46.8 | 64.8 | 89 | -28 | 120.6051181 | -132.8855422 |
| *106* | -59.4 | 22.5 | -52 | -91 | 230.7365038 | -142.9651681 |
| *107* | 18 | -3.6 | -29 | 28 | 73.42391476 | -169.4676945 |
| *108* | -14.4 | 47.7 | 31 | 21 | -177.6277228 | -151.1478108 |
| *109* | 63.9 | 72 | 81 | 1 | -250.3614717 | -122.4551667 |
| *110* | 2.7 | -32.4 | 61 | -36 | 355.4080339 | -161.2887844 |
| *111* | 80.1 | 63 | 62 | 96 | 97.55282935 | -118.7343645 |
| *112* | -86.4 | 43.2 | 97 | 23 | -145.4459623 | -122.2381778 |
| *113* | 89.1 | -25.2 | 55 | -79 | 46.62832253 | -124.8414521 |
| *114* | -63 | -39.6 | -91 | -42 | -79.69045346 | -136.3145028 |
| *115* | -61.2 | -0.9 | 24 | -54 | -106.9778034 | -144.3593447 |
| *116* | -52.2 | 27.9 | -32 | -90 | 224.6622745 | -145.5720198 |
| *117* | -33.3 | 38.7 | -95 | -13 | -154.0789383 | -150.4204301 |
| *118* | 17.1 | -34.2 | 42 | 39 | 15.543173 | -157.9562436 |
| *119* | 89.1 | -8.1 | -83 | 40 | 58.23260676 | -126.8540713 |
| *120* | 23.4 | -72.9 | 59 | -90 | 355.2876733 | -134.9833688 |
| *121* | -19.8 | -21.6 | -21 | 12 | -50.93514121 | -163.1506117 |
| *122* | -63 | 84.6 | -30 | 85 | -175.1558085 | -116.3396836 |
| *123* | 82.8 | -50.4 | -42 | 80 | 29.68079164 | -122.018969 |
| *124* | -89.1 | 55.8 | -90 | 63 | 206.2303998 | -116.5755512 |
| *125* | 67.5 | -69.3 | 8 | -43 | 15.31862651 | -122.1449497 |
| *126* | 30.6 | -62.1 | 21 | 37 | 5.979978584 | -139.4960773 |
| *127* | 19.8 | -28.8 | -96 | -33 | 24.44450666 | -159.8719673 |
| *128* | 75.6 | -1.8 | -98 | 1 | 66.41958949 | -135.567034 |
| *129* | -90 | -17.1 | -34 | -10 | 253.4965866 | -125.4772391 |
| *130* | 71.1 | -51.3 | -77 | 5 | 28.18860067 | -127.9995292 |
| *131* | -50.4 | -42.3 | -92 | 59 | -69.2013968 | -141.5846944 |
| *132* | -23.4 | 17.1 | -26 | -90 | 225.5096787 | -163.3357283 |
| *133* | -36.9 | 30.6 | 68 | 34 | -143.5358339 | -152.2640145 |
| *134* | 48.6 | 10.8 | -34 | -61 | 88.11473961 | -151.1718638 |
| *135* | -79.2 | -64.8 | 8 | 24 | -81.48480026 | -118.4515857 |
| *136* | -19.8 | 63.9 | -68 | -67 | 177.6747554 | -140.9173637 |
| *137* | 0 | -73.8 | -75 | 74 | -21.65395791 | -136.6920842 |
| *138* | -36.9 | -88.2 | -42 | 3 | -51.26024385 | -122.8851965 |

| Sıra No | Px | Py | Ex | Ey | İ $\theta_1$ | İ $\theta_2$ |
|---|---|---|---|---|---|---|
| *139* | -4.5 | 72.9 | 44 | 0 | -197.8872144 | -137.160982 |
| *140* | 87.3 | 27.9 | -57 | 47 | 80.44903545 | -125.4515977 |
| *141* | 77.4 | 61.2 | -79 | -60 | 98.77161442 | -120.8765486 |
| *142* | -76.5 | 51.3 | 7 | 42 | -151.2673972 | -125.1558862 |
| *143* | -76.5 | -62.1 | 13 | 20 | -80.4472684 | -120.9685171 |
| *144* | 11.7 | 56.7 | 68 | 89 | -208.4856884 | -146.3472085 |
| *145* | 83.7 | 76.5 | -54 | -74 | 97.88783432 | -110.9223876 |
| *146* | 33.3 | -53.1 | 0 | 98 | 13.82902068 | -143.4728587 |
| *147* | -87.3 | -24.3 | 29 | 70 | 258.6121823 | -126.1152221 |
| *148* | 81 | 16.2 | 66 | 35 | -283.0847923 | -131.2105505 |
| *149* | -70.2 | 27.9 | -94 | -16 | 226.1337386 | -135.6167753 |
| *150* | -9 | 24.3 | -3 | -70 | 192.8786449 | -165.1110162 |
| *151* | 21.6 | 43.2 | -78 | 49 | 139.4601354 | -152.0503731 |
| *152* | -15.3 | 36.9 | -56 | -92 | 190.9993325 | -156.9575337 |
| *153* | -54 | 3.6 | 52 | -24 | 250.4859913 | -148.6001323 |
| *154* | -15.3 | 56.7 | -93 | 32 | -181.9749107 | -145.8479823 |
| *155* | 42.3 | -61.2 | 10 | 92 | 12.81360719 | -136.3245826 |
| *156* | 87.3 | 67.5 | 90 | 38 | -265.7767554 | -113.0243938 |
| *157* | 71.1 | 11.7 | -24 | 83 | 78.22720909 | -137.7650744 |
| *158* | -75.6 | -47.7 | 23 | 6 | -84.29833223 | -126.9033872 |
| *159* | -81 | 48.6 | 58 | 75 | -149.1479167 | -123.6316797 |
| *160* | 75.6 | 44.1 | -62 | -40 | 94.30482175 | -128.0967692 |
| *161* | -27.9 | -36.9 | -15 | 13 | -50.46656068 | -153.2525532 |
| *162* | 54 | -20.7 | -81 | 32 | 52.21896339 | -146.3849136 |
| *163* | 84.6 | -24.3 | 61 | 57 | -312.136364 | -127.7790068 |
| *164* | 64.8 | -23.4 | -25 | -41 | 49.99485961 | -139.700148 |
| *165* | 56.7 | 17.1 | 6 | -26 | 89.55839583 | -145.5515034 |
| *166* | 16.2 | 27.9 | -40 | 8 | 140.5756129 | -161.4339969 |
| *167* | 45.9 | -60.3 | -66 | 43 | 15.01201585 | -135.4676833 |
| *168* | -12.6 | -61.2 | -53 | -98 | -29.83864433 | -143.5899793 |
| *169* | -45 | -47.7 | -68 | -43 | -62.47199395 | -141.7193372 |
| *170* | -82.8 | 73.8 | -63 | -83 | 194.6075743 | -112.6366632 |
| *171* | -48.6 | -88.2 | 73 | 31 | 300.9113403 | -119.5340031 |
| *172* | -17.1 | 63 | 61 | -19 | 176.1355809 | -141.8994873 |
| *173* | -68.4 | 79.2 | -48 | 72 | -170.7347462 | -116.9003399 |
| *174* | -3.6 | 59.4 | 73 | -90 | 166.1580223 | -145.3795862 |
| *175* | -49.5 | 51.3 | -2 | 1 | -156.9046709 | -138.2367185 |
| *176* | 47.7 | -54.9 | -99 | 12 | 19.66204543 | -137.3524423 |
| *177* | -22.5 | -73.8 | 3 | 39 | -39.64676413 | -134.6170978 |
| *178* | -51.3 | -35.1 | 55 | -1 | 286.2732974 | -143.7859054 |
| *179* | -73.8 | 1.8 | 1 | -61 | 246.9420959 | -136.6785538 |
| *180* | 67.5 | 9 | -76 | -23 | 77.68810903 | -140.1869313 |
| *181* | -72 | 18.9 | -94 | 87 | 233.4405498 | -136.2977073 |
| *182* | 70.2 | 9 | -61 | 29 | 76.58137924 | -138.5512394 |
| *183* | -57.6 | 72 | -85 | 6 | -168.7934569 | -125.0934697 |
| *184* | -3.6 | -79.2 | -69 | 34 | -25.9564335 | -133.2922574 |
| *185* | -1.8 | -48.6 | -59 | 18 | -16.1946083 | -151.8529762 |
| *186* | 79.2 | -81 | 9 | -56 | 9.854721548 | -110.9969345 |
| *187* | -36 | 49.5 | 94 | -29 | 198.2070457 | -144.3593447 |

| Sıra No | Px | Py | Ex | Ey | İ $\theta_1$ | İ $\theta_2$ |
|---|---|---|---|---|---|---|
| *188* | 81.9 | 47.7 | -87 | -79 | 91.93011862 | -123.4256893 |
| *189* | -88.2 | -6.3 | 37 | 81 | 247.8461649 | -127.5210963 |
| *190* | -53.1 | -47.7 | -31 | -48 | -68.97598241 | -138.1810062 |
| *191* | 22.5 | -70.2 | 69 | -12 | -3.857252484 | -136.7428534 |
| *192* | -30.6 | 32.4 | 35 | 80 | -149.5118566 | -154.2494409 |
| *193* | 72.9 | -55.8 | 4 | 69 | 25.24437632 | -125.3519586 |
| *194* | 11.7 | 48.6 | 73 | 100 | 151.9900344 | -151.0517815 |
| *195* | -32.4 | 71.1 | -85 | 32 | -178.4977167 | -134.0074183 |
| *196* | 90 | 54 | 29 | -75 | 89.30994071 | -116.6923684 |
| *197* | -50.4 | 9.9 | 88 | -85 | 244.0057651 | -150.2376112 |
| *198* | 46.8 | -3.6 | 89 | -24 | 72.02786877 | -152.8531482 |
| *199* | -14.4 | 67.5 | 28 | 97 | -188.1449926 | -139.6248646 |
| *200* | -34.2 | 62.1 | -83 | -88 | 188.0814986 | -138.4776769 |
| *201* | -72 | -10.8 | 42 | -99 | 257.1830217 | -137.3045122 |
| *202* | -78.3 | -70.2 | 10 | -9 | 280.1552676 | -116.5547961 |
| *203* | 27 | -86.4 | 16 | 67 | -9.556754381 | -126.178442 |
| *204* | 24.3 | 82.8 | 25 | -82 | 138.0838936 | -128.8794214 |
| *205* | 81.9 | 59.4 | -45 | -67 | 95.56364789 | -119.2224398 |
| *206* | 16.2 | -1.8 | 67 | 88 | 78.98510543 | -170.6505944 |
| *207* | 64.8 | 20.7 | -27 | -76 | 87.83101309 | -140.2304408 |
| *208* | -74.7 | -66.6 | -91 | -65 | -78.30675892 | -119.9482437 |
| *209* | -24.3 | 16.2 | -25 | 62 | -132.0866967 | -163.2067417 |
| *210* | 73.8 | -51.3 | 27 | 45 | 28.49130052 | -126.5906246 |
| *211* | -63.9 | -38.7 | -70 | -43 | -80.7327937 | -136.1333834 |
| *212* | 11.7 | -43.2 | -74 | 67 | 2.222819865 | -154.1375036 |
| *213* | -50.4 | 22.5 | -6 | 80 | -130.077265 | -147.9601689 |
| *214* | 54 | 55.8 | -41 | 36 | 113.0933027 | -134.3082235 |
| *215* | 79.2 | -19.8 | -96 | -99 | 51.8727321 | -131.8179511 |
| *216* | -58.5 | -53.1 | 10 | 33 | -71.03788715 | -133.4646573 |
| *217* | -27.9 | 78.3 | 66 | -37 | 175.054294 | -130.8844006 |
| *218* | 74.7 | -59.4 | 14 | -64 | 23.00685453 | -122.9957962 |
| *219* | -33.3 | -88.2 | -22 | 37 | -48.80827861 | -123.7516033 |
| *220* | 27.9 | -8.1 | -55 | 26 | 65.45845179 | -163.2953161 |
| *221* | 73.8 | 49.5 | 88 | 98 | 97.47138634 | -127.2407521 |
| *222* | -9.9 | 11.7 | -23 | -82 | 215.8413457 | -171.2099747 |
| *223* | 76.5 | 85.5 | 29 | -40 | 103.1754053 | -109.9911504 |
| *224* | 0.9 | 10.8 | -98 | -67 | 172.1301405 | -173.7875643 |
| *225* | 80.1 | -27 | 98 | -57 | 46.37071166 | -129.9971497 |
| *226* | -6.3 | 53.1 | 2 | -20 | 171.2588175 | -148.9852853 |
| *227* | 45.9 | 73.8 | -100 | -35 | 122.3639195 | -128.4870328 |
| *228* | 52.2 | 66.6 | -42 | -57 | 116.8810381 | -129.9396221 |
| *229* | 52.2 | -62.1 | -7 | 20 | 16.11945916 | -132.1394628 |
| *230* | -24.3 | -43.2 | 85 | 21 | -43.70671931 | -151.3020685 |

# EK-2 Yapay Sinir Ağları Test Seti ve Test Sonuçları

| | YSA Giriş Değerleri | | | | İstenen Açı Değerleri | | YSA Çıkış Değerleri | | Hata Oranları | |
|---|---|---|---|---|---|---|---|---|---|---|
| | Hedef Koordinatları (mm) | | Engel Koordinatları (mm) | | $\theta_1$ (°) | $\theta_2$ (°) | $\theta_1$ (°) | $\theta_2$ (°) | (İθ -Yθ) / Yθ | |
| **Sıra** | **Px** | **Py** | **Ex** | **Ey** | **İ $\theta_1$** | **İ $\theta_2$** | **Y $\theta_1$** | **Y $\theta_2$** | **H $\theta_1$** | **H $\theta_2$** |
| *1* | 9 | 79.2 | 78 | -95 | 150.02 | -133.02 | 150.146 | -133.04 | -0.00078 | -8.8E-05 |
| *2* | 87.3 | 22.5 | -30 | -88 | 77.65 | -126.41 | 77.3851 | -126.44 | 0.00355 | -0.00018 |
| *3* | 36 | 87.3 | 24 | -38 | 129.41 | -123.65 | 129.292 | -123.63 | 0.00095 | 0.00019 |
| *4* | 75.6 | 85.5 | 47 | 87 | -256.27 | -110.40 | -256.15 | -110.66 | 0.00051 | -0.00227 |
| *5* | 54 | 64.8 | -98 | -63 | 115.24 | -130.10 | 111.087 | -130.11 | 0.03747 | -8.9E-06 |
| *6* | 9 | 72.9 | 66 | -43 | 151.41 | -136.90 | 151.585 | -136.92 | -0.00112 | -0.00013 |
| *7* | 4.5 | 77.4 | 19 | -22 | 153.86 | -134.38 | 154.022 | -134.4 | -0.00102 | -0.00012 |
| *8* | 45.9 | 27.9 | 3 | 50 | -254.28 | -148.84 | -254.22 | -148.83 | 0.00027 | 6.8E-05 |
| *9* | 84.6 | 7.2 | 31 | -100 | 69.74 | -129.75 | 69.4923 | -129.78 | 0.00362 | -0.00013 |
| *10* | 40.5 | 16.2 | 36 | -65 | 99.20 | -154.80 | 99.2001 | -154.8 | 3.9E-05 | 4.3E-05 |
| *11* | 18 | 26.1 | 12 | 21 | -223.71 | -161.75 | -223.19 | -161.79 | 0.00233 | -0.00021 |
| *12* | 53.1 | 43.2 | -37 | 35 | 109.11 | -139.96 | 109.058 | -139.98 | 0.00052 | -5.7E-05 |
| *13* | 13.5 | 32.4 | 18 | -37 | 147.27 | -159.78 | 147.291 | -159.76 | -0.00013 | 0.00014 |
| *14* | 36.9 | 60.3 | 86 | -82 | 127.83 | -138.60 | 127.872 | -138.61 | -0.00028 | -8.6E-05 |
| *15* | 65.7 | 11.7 | -66 | 24 | 80.60 | -141.01 | 80.4686 | -141.02 | 0.00171 | -2.3E-05 |
| *16* | 0.9 | 77.4 | -57 | 40 | -203.43 | -134.46 | -202.95 | -134.5 | 0.00237 | -0.00027 |
| *17* | 86.4 | 89.1 | -38 | 60 | 97.52 | -103.28 | 97.9442 | -103.12 | -0.00429 | 0.00161 |
| *18* | 36 | 34.2 | 47 | 59 | -240.84 | -151.24 | -240.49 | -151.25 | 0.00147 | 1.6E-05 |
| *19* | 84.6 | 44.1 | 56 | 16 | -270.95 | -123.01 | -271.22 | -123.19 | -0.00098 | -0.00137 |
| *20* | 63 | 54.9 | 17 | -3 | 106.37 | -130.60 | 106.17 | -130.61 | 0.00191 | -2.5E-05 |
| *21* | 41.4 | 15.3 | -45 | -20 | 97.53 | -154.50 | -49.616 | -154.5 | -2.96577 | 4.1E-05 |
| *22* | 32.4 | 77.4 | 49 | -45 | 132.48 | -130.38 | 132.436 | -130.39 | 0.00033 | -4.8E-06 |
| *23* | 67.5 | 18.9 | 90 | -87 | 85.12 | -138.96 | 84.9686 | -138.97 | 0.00185 | -2.8E-05 |
| *24* | 6.3 | 33.3 | 61 | -58 | 159.53 | -160.48 | 159.516 | -160.46 | 9E-05 | 0.00018 |
| *25* | 73.8 | 47.7 | -69 | 13 | 96.81 | -127.87 | 96.5521 | -127.88 | 0.0027 | -5.3E-05 |
| *26* | 37.8 | 32.4 | 82 | -44 | 116.18 | -151.17 | 116.223 | -151.16 | -0.00031 | 5E-05 |
| *27* | 1.8 | 69.3 | 53 | 75 | -201.76 | -139.43 | -201.4 | -139.49 | 0.00185 | -0.00038 |
| *28* | 52.2 | 82.8 | 100 | -26 | 118.46 | -121.39 | 118.288 | -121.37 | 0.00154 | 0.00019 |
| *29* | 83.7 | 20.7 | -86 | 98 | 78.35 | -128.92 | 78.0836 | -128.94 | 0.00345 | -0.00012 |
| *30* | 18.9 | 27.9 | -37 | -69 | 136.18 | -160.59 | 136.222 | -160.59 | -0.00027 | 3.8E-05 |
| *31* | 37.8 | 65.7 | 44 | 93 | -232.18 | -135.45 | -231.5 | -135.52 | 0.00297 | -0.00047 |
| *32* | 81.9 | 77.4 | 67 | 9 | -260.91 | -111.41 | -260.92 | -111.7 | -2.9E-05 | -0.00254 |
| *33* | 0.9 | 31.5 | -39 | 22 | -190.70 | -161.86 | -190.35 | -161.9 | 0.00187 | -0.00021 |
| *34* | 30.6 | 71.1 | 46 | 92 | -226.05 | -134.46 | -225.35 | -134.52 | 0.00313 | -0.00043 |
| *35* | 2.7 | 77.4 | -39 | -65 | 155.21 | -134.43 | 155.386 | -134.45 | -0.00107 | -0.00013 |
| *36* | 75.6 | 32.4 | -80 | -95 | 88.91 | -131.43 | 88.6678 | -131.44 | 0.00279 | -5.4E-05 |
| *37* | 73.8 | 46.8 | -75 | 44 | 96.47 | -128.18 | 96.2122 | -126.19 | 0.0027 | 0.0158 |
| *38* | 80.1 | 21.6 | 97 | 92 | 80.58 | -130.98 | 80.3271 | -131 | 0.0032 | -7.7E-05 |
| *39* | 10.8 | 52.2 | 41 | -42 | 152.85 | -149.08 | 153 | -149.07 | -0.00096 | 7.6E-05 |
| *40* | 45.9 | 88.2 | -23 | -83 | 122.69 | -120.37 | 122.541 | -120.34 | 0.00127 | 0.00028 |
| *41* | 61.2 | 88.2 | -75 | -62 | 112.78 | -115.07 | 112.671 | -115.02 | 0.00097 | 0.00044 |
| *42* | 6.3 | 56.7 | 45 | 18 | -202.91 | -146.85 | -202.62 | -146.9 | 0.00143 | -0.00034 |

| **Sıra** | **Px** | **Py** | **Ex** | **Ey** | **İ $\theta_1$** | **İ $\theta_2$** | **Y $\theta_1$** | **Y $\theta_2$** | **H $\theta_1$** | **H $\theta_2$** |
|---|---|---|---|---|---|---|---|---|---|---|
| *43* | 40.5 | 41.4 | -22 | -42 | 118.79 | -146.33 | 118.843 | -146.34 | -0.00038 | -2.4E-05 |
| *44* | 9 | 32.4 | 5 | -56 | 154.79 | -160.64 | 154.797 | 21.4183 | -4.9E-06 | -8.50019 |
| *45* | 70.2 | 79.2 | -66 | -32 | 106.49 | -116.10 | 106.349 | -116.07 | 0.0014 | 0.00023 |
| *46* | 43.2 | 15.3 | -77 | 35 | 96.25 | -153.50 | 96.2427 | -153.5 | 0.00013 | 3.8E-05 |
| *47* | 15.3 | 55.8 | 66 | -84 | 147.85 | -146.36 | 148.017 | -146.37 | -0.00112 | -1.1E-05 |
| *48* | 88.2 | 13.5 | -25 | -79 | 72.20 | -127.00 | 71.9423 | -127.03 | 0.00367 | -0.0002 |
| *49* | 66.6 | 36.9 | 47 | 42 | -263.38 | -135.24 | -263.38 | -135.3 | 2E-05 | -0.00039 |
| *50* | 81 | 66.6 | -32 | -57 | 97.80 | -116.75 | 97.6285 | -116.75 | 0.00181 | 2.2E-05 |

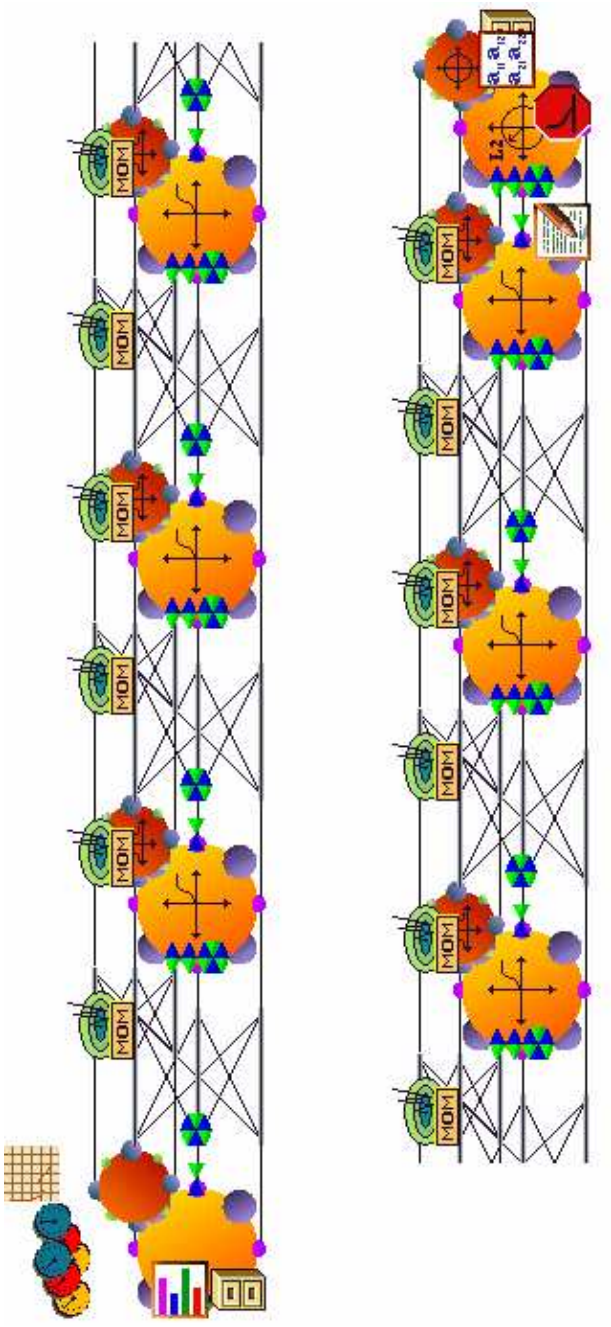
# EK-3 Bazı Trigonometrik Eşitlikler

| | | |
|---|---|---|
| 1. | $\cos\theta = a$ ise | $\theta = A\tan 2\left(\mu\sqrt{1-a^2}, a\right)$ |
| 2. | $\sin\theta = a$ ise | $\theta = A\tan 2\left(a, \mu\sqrt{1-a^2}\right)$ |
| 3. | $\cos\theta = a$ ve $\sin\theta = b$ ise | $\theta = A\tan 2(b, a)$ |
| 4. | $a\sin\theta + b\cos\theta = 0$ ise | $\theta = A\tan 2(-b, a)$<br>$\theta = A\tan 2(b, -a)$ |
| 5. | $a\sin\theta + b\cos\theta = c$ ise | $\theta = A\tan 2(a, b) + A\tan 2\left(\mu\sqrt{a^2+b^2-c^2}, c\right)$ |
| 6. | $a\cos\theta_i + b\cos\theta_j = c$<br>$a\sin\theta_i + b\sin\theta_j = d$ ise | $\theta_i = A\tan 2(d, c) + A\tan 2\left(\mu\sqrt{c^2+d^2-s^2}, s\right)$<br>$\theta_j = \theta_i + A\tan 2\left(\mu\sqrt{4a^2b^2-t^2}, t\right)$ |

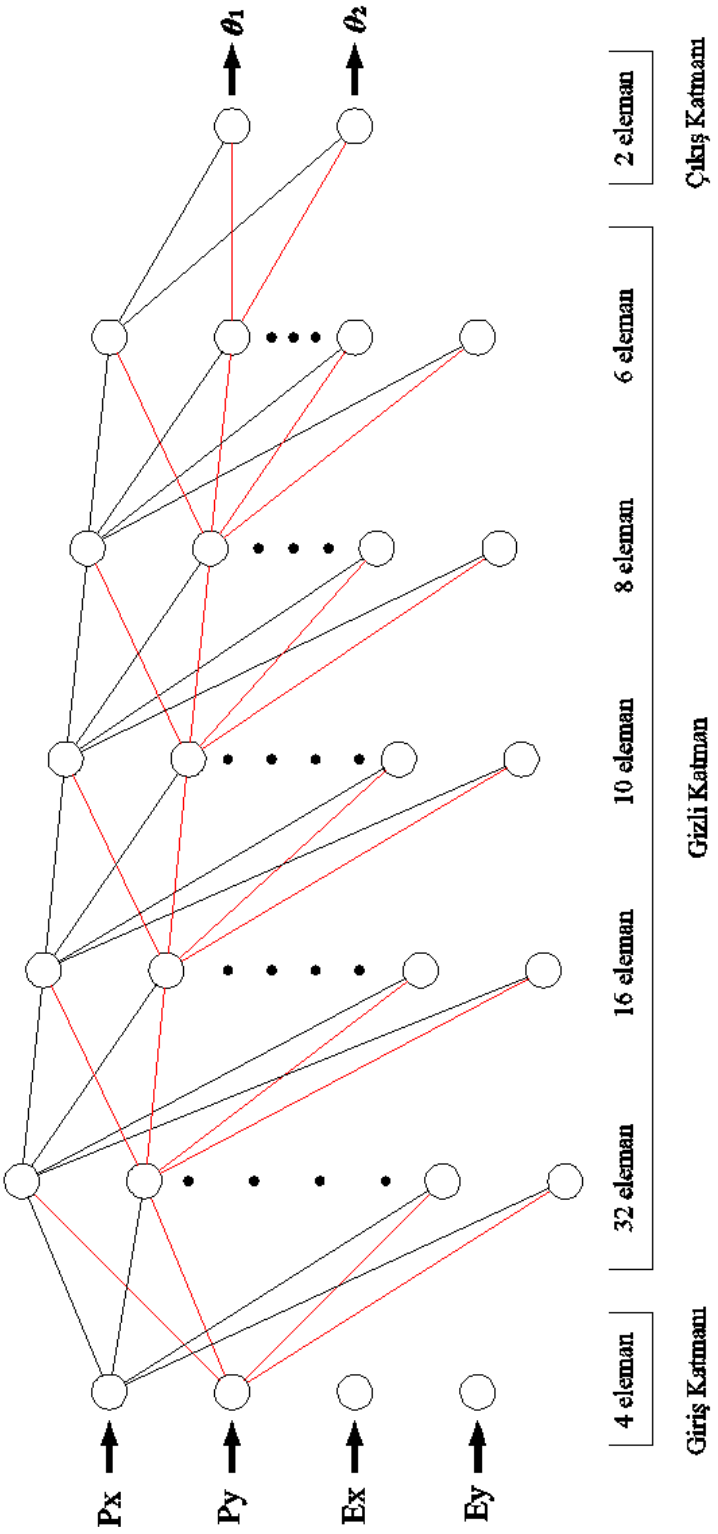
$$s = \frac{\left(a^2 - b^2 + c^2 + d^2\right)}{2} \qquad t = \left(-a^2 - b^2 + c^2 + d^2\right)$$

# EK-4 Oluşturulan Yapay Sinir Ağı

NeuroSolutions programının oluşturduğu ağın modeli çok uzun olduğundan iki parça olarak verilmiştir.



# EK-5 Oluşturulan Yapay Sinir Ağının Şematik Gösterimi



# EK-6 Deneme Eğitimi Özellikleri ve Hata Değerleri

| Deneme | Giriş Katmanı Eleman Sayısı | Gizli Katman Sayısı | Çıkış Katmanı Eleman Sayısı | Yapay Sinir Ağı | Aktivasyon Fonksiyonu | Toplam Karesel Hata |
|---|---|---|---|---|---|---|
| 1 | 4 | 1 | 2 | MLP | Tanget Hyperbolic | 0.0082 |
| 2 | 4 | 2 | 2 | MLP | Tanget Hyperbolic | 0.0061 |
| 3 | 4 | 3 | 2 | MLP | Tanget Hyperbolic | 0.0012 |
| 4 | 4 | 4 | 2 | MLP | Tanget Hyperbolic | 0.00072 |
| 5 | 4 | 5 | 2 | MLP | Tanget Hyperbolic | 0.000081 |
| 6 | 4 | 6 | 2 | MLP | Tanget Hyperbolic | 0.00012 |
| 7 | 4 | 7 | 2 | MLP | Tanget Hyperbolic | 0.00096 |
| 8 | 4 | 8 | 2 | MLP | Tanget Hyperbolic | 0.0020 |
| 9 | 4 | 1 | 2 | MLP | Sigmoid Axon | 0.00012 |
| 10 | 4 | 2 | 2 | MLP | Sigmoid Axon | 0.000095 |
| 11 | 4 | 3 | 2 | MLP | Sigmoid Axon | 0.000043 |
| 12 | 4 | 4 | 2 | MLP | Sigmoid Axon | 0.0000014 |
| **13** | **4** | **5** | **2** | **MLP** | **Sigmoid Axon** | **0.00000028** |
| 14 | 4 | 6 | 2 | MLP | Sigmoid Axon | 0.00000096 |
| 15 | 4 | 7 | 2 | MLP | Sigmoid Axon | 0.0000052 |
| 16 | 4 | 8 | 2 | MLP | Sigmoid Axon | 0.00094 |
| 17 | 4 | 9 | 2 | MLP | Sigmoid Axon | 0.0046 |
| 18 | 4 | 10 | 2 | MLP | Sigmoid Axon | 0.0067 |
| 19 | 4 | 11 | 2 | MLP | Sigmoid Axon | 0.088 |
| 20 | 4 | 1 | 2 | MLP | Linear TanH Axon | 0.075 |
| 21 | 4 | 2 | 2 | MLP | Linear TanH Axon | 0.021 |
| 22 | 4 | 3 | 2 | MLP | Linear TanH Axon | 0.0082 |
| 23 | 4 | 4 | 2 | MLP | Linear TanH Axon | 0.0065 |
| 24 | 4 | 5 | 2 | MLP | Linear TanH Axon | 0.0094 |
| 25 | 4 | 6 | 2 | MLP | Linear TanH Axon | 0.012 |
| 26 | 4 | 7 | 2 | MLP | Linear TanH Axon | 0.0086 |
| 27 | 4 | 8 | 2 | MLP | Linear TanH Axon | 0.0075 |
| 28 | 4 | 1 | 2 | MLP | Linear Sigmoid Axon | 0.0026 |
| 29 | 4 | 2 | 2 | MLP | Linear Sigmoid Axon | 0.0011 |
| 30 | 4 | 3 | 2 | MLP | Linear Sigmoid Axon | 0.00072 |
| 31 | 4 | 4 | 2 | MLP | Linear Sigmoid Axon | 0.00067 |
| 32 | 4 | 5 | 2 | MLP | Linear Sigmoid Axon | 0.00016 |
| 33 | 4 | 6 | 2 | MLP | Linear Sigmoid Axon | 0.00034 |
| 34 | 4 | 7 | 2 | MLP | Linear Sigmoid Axon | 0.00019 |
| 35 | 4 | 8 | 2 | MLP | Linear Sigmoid Axon | 0.00028 |
| 36 | 4 | 1 | 2 | MLP | Linear Axon | 0.0076 |
| 37 | 4 | 2 | 2 | MLP | Linear Axon | 0.0021 |
| 38 | 4 | 3 | 2 | MLP | Linear Axon | 0.00094 |
| 39 | 4 | 4 | 2 | MLP | Linear Axon | 0.0018 |

| | | | | | | |
|---|---|---|---|---|---|---|
| 40 | 4 | 5 | 2 | MLP | Linear Axon | 0.0057 |
| 41 | 4 | 1 | 2 | MLP | Axon | 0.013 |
| 42 | 4 | 2 | 2 | MLP | Axon | 0.0092 |
| 43 | 4 | 3 | 2 | MLP | Axon | 0.0076 |
| 44 | 4 | 4 | 2 | MLP | Axon | 0.018 |
| 45 | 4 | 5 | 2 | MLP | Axon | 0.025 |
| 46 | 4 | 1 | 2 | MLP | Bias Axon | 0.00071 |
| 47 | 4 | 2 | 2 | MLP | Bias Axon | 0.00025 |
| 48 | 4 | 3 | 2 | MLP | Bias Axon | 0.000034 |
| 49 | 4 | 4 | 2 | MLP | Bias Axon | 0.00028 |
| 50 | 4 | 5 | 2 | MLP | Bias Axon | 0.00096 |