



**ÇOK EKLEMLİ ROBOT TASARIMI VE KONTROLÜ**

**SEFA AYDIN**

**YÜKSEK LİSANS TEZİ**

**MEKATRONİK MÜHENDİSLİĞİ ANA BİLİM DALI**

**Dr. Öğr. Üyesi Levent GÖKREM**

**Aralık - 2019**

**Her hakkı saklıdır**

T.C.  
TOKAT GAZİOSMANPAŞA ÜNİVERSİTESİ  
FEN BİLİMLERİ ENSTİTÜSÜ  
MEKATRONİK MÜHENDİSLİĞİ ANA BİLİM DALI

YÜKSEK LİSANS TEZİ

ÇOK EKLEMLİ ROBOT TASARIMI VE KONTROLÜ

SEFA AYDIN

TOKAT  
Aralık - 2019

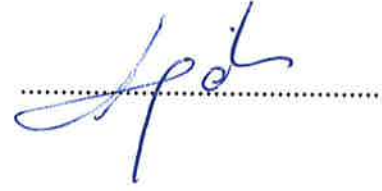
Her hakkı saklıdır

**Sefa AYDIN** tarafından hazırlanan “**Çok Eklemlili Robot Tasarımı ve Kontrolü**” adlı tez çalışmasının savunma sınavı 26 ARALIK 2019 tarihinde yapılmış olup aşağıda verilen Jüri tarafından Oy Birliđi / Oy Çokluđu ile Gaziosmanpaşa Üniversitesi Fen Bilimleri Enstitüsü MEKATRONİK MÜHENDİSLİĐİ ANA BİLİM DALI'nda YÜKSEK LİSANS TEZİ olarak kabul edilmiştir.

Jüri Üyeleri

İmza

Danışman  
Dr. Öğr. Üyesi Levent GÖKREM  
Gaziosmanpaşa Üniversitesi



İkinci Danışman  
Dr. Öğr. Üyesi Mehmet Serhat CAN  
Gaziosmanpaşa Üniversitesi



Dr. Öğr. Üyesi Mahmut DURGUN  
Gaziosmanpaşa Üniversitesi



Doç. Dr. Ünal KURT  
Amasya Üniversitesi



ONAY

Prof. Dr. Cetin ÇEKİC  
Fen Bilimleri Enstitüsü Müdürü  
26/12/2019



## TEZ BEYANI

Tez yazım kurallarına uygun olarak hazırlanan bu tezin yazılmasında bilimsel ahlak kurallarına uyulduğunu, başkalarının eserlerinden yararlanılması durumunda bilimsel normlara uygun olarak atıfta bulunulduğunu, tezin içerdiği yenilik ve sonuçların başka bir yerden alınmadığını, kullanılan verilerde herhangi bir tahrifat yapılmadığını, tezin herhangi bir kısmının bu üniversite veya başka bir üniversitedeki başka bir tez çalışması olarak sunulmadığını beyan ederim.



**SEFA AYDIN**

**26 Aralık 2019**

## ÖZET

### YÜKSEK LİSANS TEZİ

#### ÇOK EKLEMLİ ROBOT TASARIMI VE KONTROLÜ

SEFA AYDIN

TOKAT GAZİOSMANPAŞA ÜNİVERSİTESİ FEN BİLİMLERİ ENSTİTÜSÜ

MEKATRONİK MÜHENDİSLİĞİ ANA BİLİM DALI

I. TEZ DANIŞMANI: DR. ÖĞR. ÜYESİ LEVENT GÖKREM

II. TEZ DANIŞMANI: DR. ÖĞR. ÜYESİ MEHMET SERHAT CAN

Günümüz dünyasında robot kavramı hayatımızın birçok alanına girmiş bulunmaktadır. Teknolojinin gelişmesi ile birlikte sistemlerin otonom kontrolü, üretim sistemlerinin gelişimi ve bu sistemlerin üretim gücü günden güne gelişmekte ve değişmektedir. Bu gelişim ve değişimin en önemli faktörleri şüphesiz sistemlerde kullanılan robotlardır. Bu robotlar endüstri sanayisi başta olmak üzere manüel veya otonom kontrol gerektiren birçok sektörde kullanılmaktadır. Otonom kontrollü çalışan tip robotlar herhangi bir dış etki olmadan işlem yapabilme, görüntü alabilme özelliğine sahiptirler. Sera çiftçiliği alanında doğru zamanda ilaçlama, sulama veya toplama gibi faktörlerin dışarıdan bir müdahale olmadan otonom kontrol edilebilmesi, insan sağlığının riske girebileceği bölgelerden görüntü aktarımı gibi alanlarda kullanılmak üzere bu tez çalışmasında 6 bacaklı bir örümcek robotun 3D tasarımı ve Matlab tabanında kontrolü yapılmıştır. Tasarım aşamasında Solidworks tasarım programı, Simulink aşamasında MatlabR2017b ve kütüphanesi kullanılmıştır. Örümceklerin anatomik yapısından farklı olarak denge faktörünü maksimuma çıkarmak amacı ile bacak yapıları 60 derece aralıklı olacak şekilde tasarlanmıştır. Görüntü işleme özelliği için uzayan boyun yapısı ile birlikte kamera şaseninin tam ortasına yerleştirilmiştir. Robotun kontrolü ise simulink ortamında ayrı ayrı olacak biçimde PID ve Fuzzy kontrol ile yapılmıştır. İnterpolasyon tekniği ile robotun X-Y koordinat sisteminde istenilen koordinatlara istenilen sürede gönderimi sağlanmıştır. Örümcek robotun tüm parçalarının ağırlıkları, gerçek parça ağırlık değerleri ile aynı olarak girilmiştir. Düz zeminde hareket, eğimli zeminde hareket, farklı ağırlıklar ve farklı sürtünme kat sayılarında hareket olarak kontroller her iki kontrol tipi içinde yapılarak değerler alınmıştır. Alınan değerlerden yola çıkılarak Fuzzy ile PID kontrol tipleri karşılaştırılmıştır.

2019, 107 SAYFA

**ANAHTAR KELİMELELER:** 3DTasarım, Solidworks, Matlab, Simulink, PID, Fuzzy

## **ABSTRACT**

### **MASTER THESIS**

#### **MULTI AXIS ARTICULATED ROBOT DESIGN AND CONTROL**

**SEFA AYDIN**

**TOKAT GAZIOSMANPASA UNIVERSITY**

**GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES**

**DEPARTMENT OF MECHATRONICS ENGINEERING**

**I.SUPERVISOR:ASST. PROF. DR. LEVENT GÖKREM**

**II. SUPERVISOR: ASST. PROF. DR MEHMET SERHAT CAN**

In today's world, the concept of gir robot has entered many areas of our life. With the development of technology, autonomous control of systems, the development of production systems and the production power of these systems are developing and changing day by day. The most important factors of this development and change are undoubtedly the robots used in the systems. These robots are used in many industries that require manual or autonomous control, especially in the industrial area. Autonomously controlled type robots have the ability to process without any external effect and to take images. Since they are stable in structure and can work without interruption, the margin of error is minimized and quality results are achieved. In this thesis, 3D design of a 6-legged spider robot and its control in Matlab base were performed in order to be able to control the factors such as spraying, irrigation or collection at the right time in the field of greenhouse farming without external intervention. In the design phase, the Solidworks design program, the Simulink stage, and the library MatlabR2017b were used. Unlike the anatomical structure of spiders, the leg structure is designed to be 60 degrees apart to maximize the equilibrium factor. The camera is positioned in the center of the chassis with the extended neck structure for image processing. The control of the robot was made with PID and Fuzzy control in the simulink environment. The interpolation technique was used to send the robot to the desired coordinates in the X-Y coordinate system. The weights of all parts of the spider robot are entered as the same as the actual parts weight values. The values were taken in both control types by taking motion on flat ground, movement on inclined ground, different weights and different friction coefficients. Based on the values taken, Fuzzy and PID control types were compared.

2019, 107 PAGE

**KEYWORDS:** 3D Design, Solidworks, Matlab,Simulation, PID, Fuzzy

## ÖNSÖZ

Yüksek lisans eğitimim boyunca yardımını esirgemeyen tez seçimimde, çalışmalarım da sürekli destekçim olan danışmanım Dr. Öğr. Üyesi Levent GÖKREM hocama, bilgi birikimini benden esirgemeyen çalışmalarım da büyük emeği olan ikinci danışmanım Dr. Öğr. Üyesi Mehmet Serhat CAN hocama teşekkürü borç bilirim. Fikirleri ile sürekli destek olan Bülent TURAN hocama da ayrıca teşekkür ederim.

Sadece yüksek lisans eğitimimde değil tüm eğitim hayatım boyunca yanımda olan ve desteğini biran olsun esirgemeyen anneme ve babama sonsuz teşekkür ederim. Ayrıca yüksek lisans eğitimimde yardımlarını esirgemeyen yüksek lisans öğrencisi Elif YÜCEL'e teşekkür ederim.

## İÇİNDEKİLER

### Sayfa

ÖZET .....	i
ABSTRACT .....	ii
ÖNSÖZ.....	iii
İÇİNDEKİLER.....	iv
ŞEKİL LİSTESİ .....	vi
GRAFİK LİSTESİ.....	x
ÇİZELGE LİSTESİ .....	xi
1. GİRİŞ.....	1
2. EKLEMLİ ROBOTLAR .....	8
2.1. Neden Eklemlili Yapı ?.....	8
2.2. Eklemlili Yapının Çeşitleri ve Gelişimi.....	10
2.2.1. Tek bacaklı robotlar.....	10
2.2.2. İki bacaklı robotlar.....	11
2.2.3. Üç bacaklı robotlar .....	12
2.2.4. Dört bacaklı robotlar.....	12
2.2.5. Beş bacaklı robotlar .....	13
2.2.6. Altı bacaklı robotlar.....	13
2.3. Hexapod Robotların Yürüyüş Şekilleri .....	15
2.3.1. Tripod yürüyüş şekli .....	16
2.3.2. Wave yürüyüş şekli .....	17
2.3.3. Ripple yürüyüş şekli .....	17
3. ROBOT KİNEMATİĞİ.....	18
3.1. İleri Yön Kinematığı.....	19
3.2. Ters Yön Kinematığı .....	19
4. MATERİYAL .....	21
4.1. Solidworks.....	21
4.1.1.Solidworks ile Matlab arasında veri aktarımı.....	22
4.2. Matlab .....	26
4.3. PID Kontrolör .....	27
4.3.1. Proportional .....	29
4.3.2. Integral .....	29
4.3.3. Derivative .....	29
4.4.Fuzzy Kontrolör.....	30
5. YÖNTEM .....	32
5.1.Tasarım Aşaması .....	32
5.2. Simulink Kontrol Aşaması .....	46
5.2.1. Xml dosyası oluşturma .....	46
5.2.2. Montaj çağırma işlemi .....	47



5.2.3. Çözücü(solver) seçimi .....	49
5.2.4. Blok oluşturma .....	51
5.2.5. Simulinkte çağırılan montajın parametre ayarı .....	53
5.2.6. Bacak eklemlerine pozisyon ve sınırlama yapma işlemi.....	55
5.2.7. Mekanik parametreler ve uzay ortamında konum belirleme .....	56
5.2.8. Robot konum kontrolü.....	57
5.2.9. Bacak kontrol bloğu.....	58
5.2.10. Uzay boşluğunda X-Y-Z eksenli düzlem tanımlama .....	61
5.2.11. Sürtünme ve yüzey tepki kuvveti tanımlama .....	62
5.2.12. Sinyal giriş bloğu oluşturma.....	65
5.2.13. Yörünge oluşturma işlemi .....	66
5.2.14. Yüzey ile robot arasında sürtünme tanımlama .....	67
5.2.15. PID kontrol .....	71
5.2.16. Fuzzy kontrol.....	78
5.2.17. Eğimli yol oluşturma .....	87
5.2.18. Servo motorların tork grafikleri.....	88
6. SONUÇ.....	91
7. ÖNERİLER .....	92
8. KAYNAKLAR.....	93
9. EKLER .....	95
10. ÖZGEÇMİŞ .....	107

## ŞEKİL LİSTESİ

<u>Şekil</u>	<u>Sayfa</u>
Şekil 1.1. İnsan yürüyüşünün tekerlekli yürüyüşe benzetilmesi.....	2
Şekil 1.2. Stanford Cart tekerlekli mobil robot . ....	2
Şekil 1.3. Shakey mobil robot .....	3
Şekil 1.4. RB5X ve HERO mobil robotlar. ....	4
Şekil 1.5. RHEX Robotlar. ....	4
Şekil 1.6. Oyuncak robot Aibo .....	5
Şekil 1.7. Sojourner ve Pathfinder robotları. ....	5
Şekil 1.8. İki bacaklı Humanoid robot ASIMO.....	6
Şekil 1.9. İnsan görünümlü robot Geminoid F .....	6
Şekil 1.10. İki bacaklı robot Atlas. ....	7
Şekil 1.11. İnsansı robot Sofya. ....	7
Şekil 2.1. Örnek bir hexapod yapılı robot .....	9
Şekil 2.2. Eklemlili bacak yapısına sahip örnek robot görüntüleri .....	10
Şekil 2.3. Tekerlekli bacak yapısına sahip örnek robot görüntüleri .....	10
Şekil 2.4. Tek bacaklı robotlar.....	11
Şekil 2.5. QRIO ve ASIMO humanoid robotlar .....	10
Şekil 2.6. Üç bacaklı STRiDer robotlar .....	12
Şekil 2.7. Dört bacaklı robotlar .....	13
Şekil 2.8. Beş bacaklı denizyıldızı robotlar .....	13
Şekil 2.9. Altı bacaklı örümcek robotlar.....	14
Şekil 2.10. Altıgen ve dikdörtgen düzende hexapod yapı .....	15
Şekil 3.1. Hexapod örümcek robotun bacağının CAD görünümü.....	18
Şekil 3.2. Eklemlili yapı üzerine koordinat sistemi yerleştirilmesi .....	18
Şekil 4.1. Tasarım ve kontrol blok diyagramı .....	21
Şekil 4.2. Solidworks panel anlatımı için örnek görüntü .....	22
Şekil 4.3. Solidworks ile Matlab ilişkilendirme dosyaları.....	23
Şekil 4.4. Eklenti dosyasına yol ekleme .....	24
Şekil 4.5. Eklenti dosyasını Matlab içerisine yükleme.....	24
Şekil 4.6. Matlab'in otomasyon sunucusu olarak kayıt edilmesi .....	25
Şekil 4.7. Autodesk Inventor uygulamasının etkinleştirilmesi.....	25
Şekil 4.8. Solidworks ortamında montajın sldprt uzantılı kaydedilmesi .....	26
Şekil 4.9. Matlab panel gösterimi .....	27
Şekil 4.10. PID kontrolör temel şeması .....	28
Şekil 4.11. P-I-D kontrolör blok diyagramı.....	28
Şekil 4.12. P-I-D değerlerinin sinyale katkısı.....	30
Şekil 4.13. Bulanık sistem mimarisi .....	30
Şekil 4.14. Bulanık mantık mimarisi .....	31

Şekil 5.0. Yöntem şeması .....	32
Şekil 5.1. Şasi ön çizim .....	32
Şekil 5.2. Şasi Cad taslak çizimi.....	33
Şekil 5.3. Bacak hareket aralığı .....	33
Şekil 5.4. Şasi Cad çizimi .....	33
Şekil 5.5. Güç kutusu çizimi.....	34
Şekil 5.6. Şasinin Son hali .....	34
Şekil 5.7. Güç kutusu taslak ve ekstrüzyon .....	34
Şekil 5.8. Güç kutusu.....	35
Şekil 5.9. Şasi giydirme taslak çizimi.....	35
Şekil 5.10. Güç kutusu bölmesi .....	36
Şekil 5.11. Güç kutusu bölmesi yapımı .....	36
Şekil 5.12. Yan kanatçık tasarımı .....	37
Şekil 5.13. Şasi giydirmenin son hali .....	37
Şekil 5.14. Anten taslak çizimi .....	38
Şekil 5.15. Antene üç boyut kazandırma işlemi .....	38
Şekil 5.16. Anten çizimi .....	39
Şekil 5.17. Kamera taslak çizim .....	39
Şekil 5.18. Kamera mercekle tasarımı.....	40
Şekil 5.19. Kamera tasarımı .....	40
Şekil 5.20. Kamera tutacağı tasarım .....	40
Şekil 5.21. Kamera montaj hali .....	41
Şekil 5.22. Pabucun kesit görüntüsü.....	41
Şekil 5.23. Pabuç tasarımı .....	41
Şekil 5.24. Servo motor taslak çizimi.....	42
Şekil 5.25. Servo motor mil çizimi.....	42
Şekil 5.26. Servo motor .....	43
Şekil 5.27. Servo-1 bacak aparatı taslak çizimi.....	43
Şekil 5.28. Servo-1 bacak aparatı .....	43
Şekil 5.29. Aparat-2 Taslak çizim .....	44
Şekil 5.30. Servo-2 bacak aparatı .....	44
Şekil 5.31. Örümcek robot için tasarlanan parçalar.....	45
Şekil 5.32. Örümcek robot tasarımı.....	45
Şekil 5.35. Xml dosyası oluşturma .....	46
Şekil 5.36. Jenerasyon hatası .....	47
Şekil 5.37. Montaj çağırma işlemi .....	47
Şekil 5.38. Tasarımın temel Matlab kontrol paneli görünümü.....	48
Şekil 5.39. Matlab tarafından oluşturulan düzenlenmemiş ilk temel devre .....	49
Şekil 5.40. Simulink çözücü seçimi.....	50
Şekil 5.41. Hexapod robotun 1 numaralı bacak bloğu .....	51
Şekil 5.42. Hexapod bacak blokları.....	51
Şekil 5.43. Hexapod robotun bacak bloğu.....	52

Şekil 5.44. Hexapod robot gövde yapıları bloğu .....	52
Şekil 5.45. Hexapod robotun gövde bloğu .....	53
Şekil 5.46. Çağırma sonrası ilk görünüm .....	53
Şekil 5.47. Eklem hareket kontrol girişi ayarı .....	54
Şekil 5.48. Eklem kontrol pini.....	54
Şekil 5.49. Bacaklar arası açılı aralıkları .....	55
Şekil 5.50. Tasarım üzerinde bir bacağın servo motorları.....	56
Şekil 5.51. Robotun bacak başlangıç pozisyonları .....	56
Şekil 5.52. Mekanik parametreler.....	57
Şekil 5.53. X-Y ve Trajectory Scopeleri .....	57
Şekil 5.54. Robot konum takibi bloğu.....	58
Şekil 5.55. Bacak kontrol bloğu .....	58
Şekil 5.56. Bacak kontrol bloğu iç görünümü .....	59
Şekil 5.57. Birinci bacağın kontrol bloğu.....	59
Şekil 5.58. İki numaralı bacağına ait kontrol bloğu.....	60
Şekil 5.59. Periyotlar arasına gecikme süresinin uygulanması .....	60
Şekil 5.60. İkinci eklemlerin giriş sinyali .....	61
Şekil 5.61. Yol oluşumu ve parametre ayarları .....	61
Şekil 5.62. Transform bloğu parametreleri.....	62
Şekil 5.63. Robot ile yüzey arasında sürtünme oluşturma .....	63
Şekil 5.64. Her bacak için kullanılan pabuç bloğu.....	63
Şekil 5.65. Pabuçlara sürtünme noktası belirlemek.....	64
Şekil 5.66. Pabuçlara sürtünme çemberi eklemek .....	64
Şekil 5.67. Robotun yüzeye tutunamaması olayı .....	65
Şekil 5.68. Sinyal giriş Bloğu.....	65
Şekil 5.69. Sinyal giriş bloğu içeriği .....	66
Şekil 5.70. Trajectory oluşturma .....	67
Şekil 5.71. Yörünge oluşturma .....	67
Şekil 5.72. Noktalar arası zaman farkı oluşturma işlemi.....	68
Şekil 5.73. Yörünge grafiği .....	68
Şekil 5.74. Statik sürtünme tanımlama.....	69
Şekil 5.75. Oluşturulan XML dosya verilerinde kütle değişimi.....	70
Şekil 5.76. Robotun Yüzeyde tutunamaması.....	71
Şekil 5.77. PID kontrolör bloğu.....	71
Şekil 5.78. PID kontrolör bloğunun iç görünümü .....	72
Şekil 5.79. PID kontrolör parametre değerleri .....	73
Şekil 5.80. Fuzzy blok tasarımı .....	78
Şekil 5.81. Fuzzy simulink bloğu .....	78
Şekil 5.82. Fuzzy simulink bloğu içeriği .....	79
Şekil 5.83. Fuzzy girişleri.....	79
Şekil 5.84. Fuzzy çıkışı.....	80
Şekil 5.85. Fuzzy kontrol için yazılan kurallar.....	80

Şekil 5.86. Matlab ortamında eğimli yol oluşturma .....	87
Şekil 5.87. Matlab ortamında eğimli yol oluşturma .....	88
Şekil 5.88. Servo motorların tork değerlerini okuma .....	89

## GRAFİK LİSTESİ

<u>Grafik</u>	<u>Sayfa</u>
Grafik 5.1. Y ekseninde izlenecek olan referans sinyali .....	73
Grafik 5.2. Y ekseninde referans yörüngeyi takip sinyali .....	74
Grafik 5.3. Y eksenindeki referans ve takip sinyali .....	74
Grafik 5.4. Y eksenindeki referans ve takip sinyali hatası .....	75
Grafik 5.5. X ekseninde takip edilecek olan referans sinyali .....	75
Grafik 5.6. X eksenindeki takip sinyali .....	76
Grafik 5.7. X eksenindeki referans ve takip sinyali. ....	76
Grafik 5.8. X eksenindeki referans ve takip sinyallerinin 0-2,3ms aralığı. ....	77
Grafik 5.9. PID kontrolde X eksenindeki hata. ....	77
Grafik 5.10. X ekseninde takip edilecek olan referans sinyali. ....	82
Grafik 5.11. X eksenindeki takip sinyali. ....	82
Grafik 5.12. X eksenindeki referans ve takip sinyali. ....	83
Grafik 5.13. X eksenindeki referans-takip sinyalinin 25-29sn aralığı. ....	83
Grafik 5.14. Fuzzy kontrolde X eksenindeki hata. ....	84
Grafik 5.15. Y eksenine için Trajectory ile oluşturulan takip sinyali. ....	84
Grafik 5.16. Y eksenindeki takip sinyali. ....	85
Grafik 5.17. Y ekseninde yapılan referans ve takip sinyalleri. ....	85
Grafik 5.18. Y ekseninde referans ve takip sinyallerinin 42-52sn aralığı. ....	86
Grafik 5.19. Fuzzy kontrol ile Y ekseninde referans takip hatası. ....	86
Grafik 5.20. Eğimli yolda elde edilen takip grafiği. ....	88
Grafik 5.21. Servo-1 tork grafiği. ....	89
Grafik 5.22. Servo-2 tork grafiği. ....	90

## ÇİZELGE LİSTESİ

<u>Çizelge</u>	<u>Sayfa</u>
Tablo 1.1. Hexapod robot için Tripod, Wave ve Ripple yürüyüş .....	16
Tablo 3.1. Denavit-Hartenberg değişken tablosu .....	19
Tablo 5.1. Bazı Cisimler Arası Statik Sürtünme Katsayıları.....	69

## 1. GİRİŞ

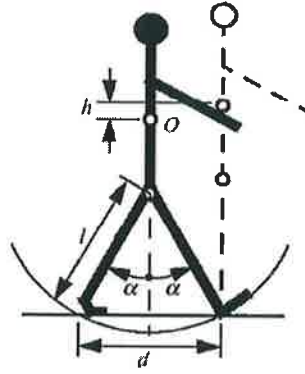
Robot kelimesi tarihte ilk olarak 1921'de Karel Capek tarafından bir oyun yazısında kullanılmıştır. Kökeni Çek diline dayanan bu kelime Çek dilinde esir anlamına gelmektedir. Esirlerden ilham alınarak yapılan bu benzetme, insanoğlunun istediği her işi yaptırabileceği ama kendi yapıtı olacak olan icatları kastediyordu. Bu oyun zamanla insanların yapabildiği tüm işleri robotların yapabileceğini ve en sonunda robotların dünyanın yönetimini ele geçireceklerini anlatıyordu. Bu başlangıç ile birlikte robot terimi tüm dünyada genel bir terim olarak kullanılmaya başlanmıştır.

Dünyada var olan her robot için geçerliliği kabul edilen 3 temel yasa bulunmaktadır. Bu yasaları yine bir kurgu yazarı olan Isaac Asimov geliştirmiştir. Bu 3 temel yasa robotların özgürlüğünü insan yararına kısıtlamaktadır.

1. Kullanıma hazır bir robot, kesinlikle insanlara zarar veremez veya insanların zarar görmesine seyirci kalmaz.
2. Kullanıma hazır bir robot, verilen emirleri madde bir ile çelişmediği sürece kesinlikle uygulamak zorundadır.
3. Kullanıma hazır bir robot, ilk iki maddeyle çelişmedikçe kendi varlığı için çabalamak zorundadır(Isaac, 1950)

Bir yol üzerinde robotun hareket edebilmesi için yüzeye göre farklılık gösteren hareket mekanizmalarına ihtiyacı vardır. Bu mekanizmalara; genellikle düz yollarda kullanılan tekerlekli yapılar, eğimli arazilerde kullanılan eklemli yapılar ve havada hareket sağlayan kanatçık yapılar örnek olarak verilebilir. Bu hareket şekillerinin merkez noktası eklemli ve tekerlekli yapılardır. Eklem ayaklı yapılar doğadaki örümceklerden ilham alınarak tasarlanmıştır. Bu ilhamın sebebi zorlu koşullarda başarılı hareket edebilme yeteneğidir. Günümüzdeki eklem yapılı mobil robotların ayak sayıları dört ve altı sayılarında ağırlık göstermiştir. Bu ayak yapılarında denge faktörü de ön plana çıkmaktadır. Eklem yapılı hareket sisteminin kullanımındaki en temel problemler kontrol bulanıklığı, mekanik karmaşa ve aşırı güç tüketimidir. En yaygın olarak kullanılan diğer hareket sistemi ise tekerlekli hareket sistemidir. Günümüzde son derece yaygın olan bu hareket sistemini doğada benzetebileceğimiz hiç bir yapı yoktur. İnsan yürüyüşünüŞekil1.1'de görüldüğü gibi uzun vadede inceleyecek olursak tekerlekli yapıya benzetilebilir(Böttcher, 2004).



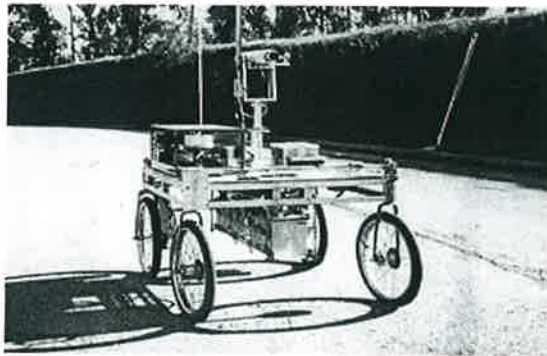


Şekil 1.1. İnsan yürüyüşünün tekerlekli harekete benzetilmesi

Buhar motorları ve yanmalı motorların icadı ile birlikte tekerlek yapılarına ihtiyaç duyulmuştur. Tekerlekli yapılar kullanılarak uzak mesafelere trenler ile ulaşım oldukça kolaylaşmıştır. Fakat engebeli ve pürüzlü yüzeylere tekerlekli yapılar uygun değildir. Bu sorun zamanla eklem bacaklı yapıların doğadan esinlenerek gelişmesine olanak sağlamıştır. Eklem bacaklı hareket büyük oranda doğadaki hayvanları taklit ederek geliştirilen alternatif bir harekettir(Mahajan ve ark., 1997).

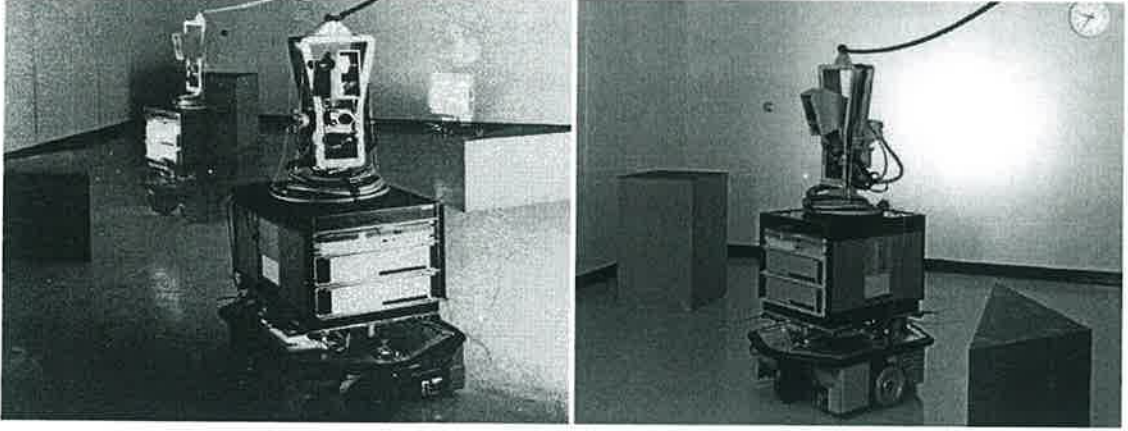
Robot ve robotun hareketi arasındaki en önemli faktör arazi şartlarıdır. Pürüzlü ve eğimli yüzey, sürtünme, sertlik ve farklı yüzey koşullarına sahip araziler Hardarson'un sıraladığı temel özelliklerdir(Hardarson,1998).

Robotların tarihçesi çok eskiye dayansa da mobil robotlar üzerine yapılan ciddi çalışmaların başlangıcı olarak Stanford Cart'ın yaptığı çalışma gösterilmektedir. Cart'ın 1970'de yaptığı bu mobil robot ana hatlarıyla: bir bilgisayar, üzerinde bulunan kamera ve besleme hattını oluşturan iki büyük akü den oluşuyordu. Cart'ın robotu engelleri algılayabilme, parkurları başarılı bir şekilde tamamlayabilme ve dolaştığı bölgeyi haritalayabilme özelliklerine sahipti. Şekil 1.2'de görülen Cart'ın mobil robotu üzerindeki ana bilgisayar ile işlemleri yaparak, çizgi takip edebilme özelliği olan ilk mobil robot olarak tarihe geçmiştir(Earnest, 1980).



Şekil 1.2. Stanford Cart tekerlekli mobil robot

1972 yılında Stanford Enstitüsü tarafından radyo frekansları ile genel komutlar verilerek kontrol edilen Shakey isimli robot geliştirildi. Shakey adını hareketi sırasındaki titreşimli yürüyüşünden alıyordu. Cart'ın robotuna göre daha küçük boyuta sahip olan Shakey üzerinde kamera ve çarpma sensörleri bulunuyordu bu da robotun kendisine verilen görevleri yerine getirebilmek için gereken yolları kolaylıkla bulabileceği anlamını taşıyordu. Robotun engelleri algılaması üzerine yapılan testler Şekil 1.3'de görülmektedir(Anonim).



Şekil 1.3. Shakey mobil robot

1977'de sinemalarda yayınlanan Yıldız Savaşları filmi, içerisinde kullanılan animasyon robotlar ile robot kelimesinin insan hayatında yayılmasında önemli rol oynamıştır(Anonim).

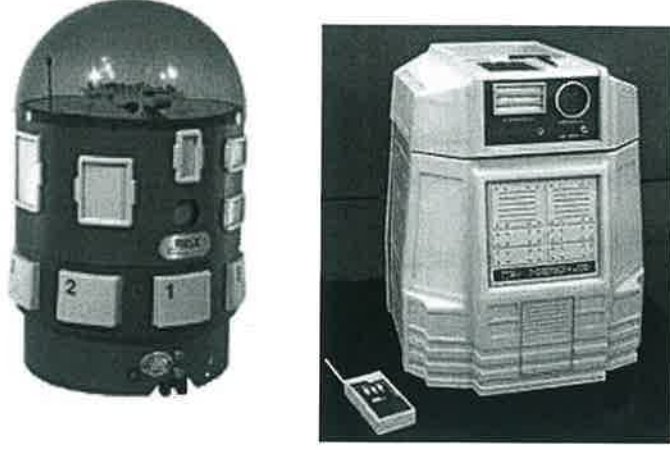
1978 yılında PUMA adı verilen mobil robot tasarlanmış ve piyasada satılmaya başlanmıştır(Anonim).

1979 yılında Cart'ın robotu geliştirildi. Üzerindeki kameradan aldığı görüntüler farklı bir algoritma uygulanarak işlendi. Bu algoritmayı kullanan robot engelli parkur haline getirilmiş bir odanın bir kapısından girip diğer kapısından başarılı bir şekilde geçmiştir(Anonim).

1980 yılına gelindiğinde halkın mobil robotlara ilgisi iyice artmış ve bu duruma binaen talepler başlamıştır. Talepleri karşılamak için ev ve iş yerlerinde kullanmak için üretilen mobil robotlar ortaya çıkmaya başlamıştır. Bu robotlara örnek olarak Şekil 1.4'de gösterilmiş olan RB5X ve HERO robot serileri verilebilir (Anonim).

Stanford Enstitüsünün çalışmaları sonucunda 1984'de Shakey'den daha gelişmiş olan Lakey mobil robotu üretildi. Lakey'in üretimi ile mobil robotlara olan ilgi daha da artmıştır(Anonim).

Mobil robotlara olan ilginin artışı ile birlikte 1990 yılında Amerika Birleşik Devletlerinde 10 dolaylarında, Japonya'da ise 40 mobil robot firması kuruldu(Anonim).

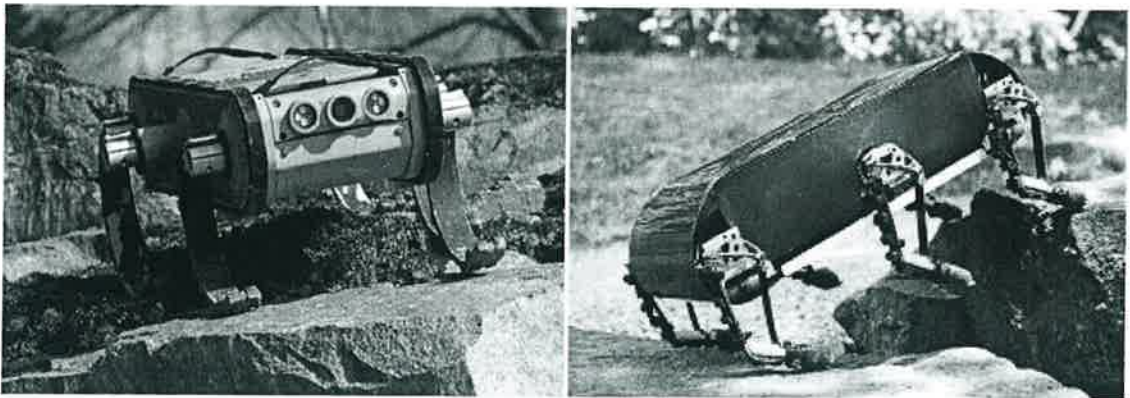


Şekil 1.4. RB5X ve HERO mobil robotlar

1993 ve 1994 yıllarında Geçmişte üretilen mobil robotlara nazaran fiyatı daha uygun olan ERRATIC ve PIONEER1 isimli mobil robotlar üretildi ve piyasaya sürüldü(Anonim).

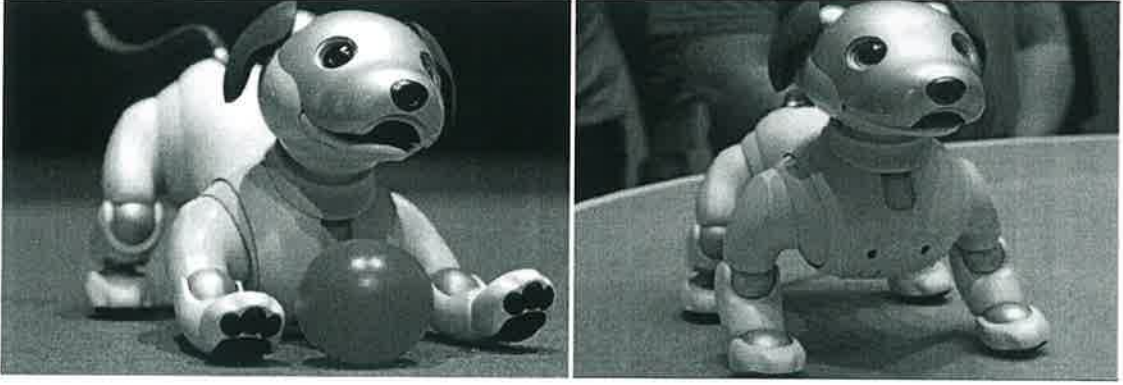
1998 yılında oyuncak mobil robot FURBY üretildi ve piyasaya sürüldü(Anonim).

Mobil robotlardaki en önemli gelişmelerden bir tanesi RHex robottur. Boston Dynamics tarafından 1999 yılında geliştirilen bu robot yarım daire şeklinde altı bacağı sahiptir. Bacakların yarım daire olmasındaki sebep engebeli alanlarda robotun gövdesinin yere yakınlığını koruyarak askıda kalmayı önleyerek hareketi sürdürebilmektir. Şekil 1.5'de görülenRHex robot uygulanan testlerde merdiven çıkabilme, yamaçlara tırmanabilme gibi birçok zorlu işi başarılı bir şekilde yapmıştır(Buehler ve ark., 2001).



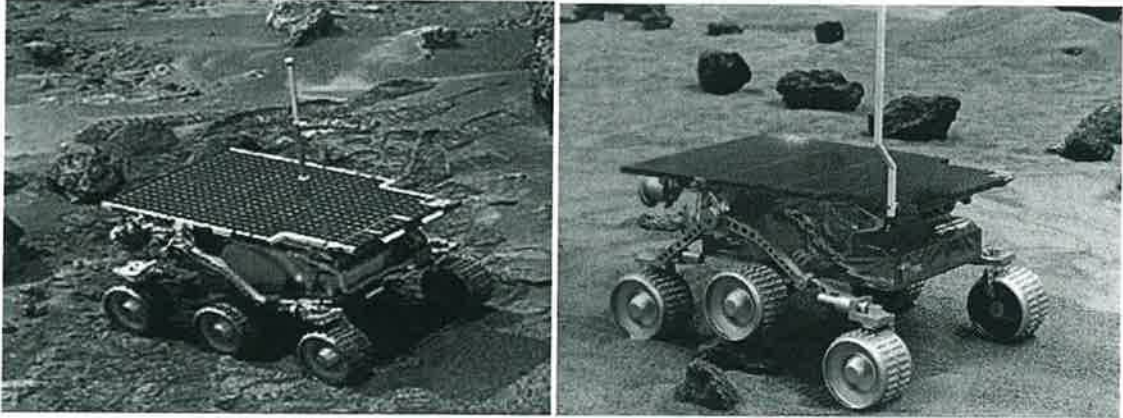
Şekil 1.5. RHEX robotlar

Robotların insan hayatında daha yaygın hale gelmesi için Şekil 1.6'da görülen ev hayvanı görünümlü mobil robot SONY ve Aibo üretildi(Anonim).



Şekil 1.6. Oyuncak robot Aibo

NASA'nın gezegen ve uyduları araştırmak amacıyla yaptığı çalışmalar sonucunda geliştirilen Şekil 1.7'de gösterilmiş olan Sojourner ve Pathfinder isimli robotlar 1996 yılında Mars'a gönderilmiştir. Gönderilen bu gezici mobil robotlar gezegenlerin yüzeylerini inceliyor ve görüntüler alıyordu. Robotların iki tane olarak gönderilmesinin bir amacı vardı. Sojourner isimli robot çevreden gelebilecek zararlara karşı savunma sistemleri ile donatılmıştı. Buda Sojourner'in hakkında bilgi sahibi olunmayan yüzeylerde kendisine yol açarak ilerlemesine olanak sağlıyordu(Anonim).



Şekil 1.7. Sojourner ve Pathfinder robotları

İnsan görünümlü robotlarda önemli bir ilerleme olan Asimo'yu Honda 2000 yılında dünyaya duyurdu. Şekil 1.8'de görüntüsü verilmiş olan ASIMO 2000 yılına kadar yapılmış olan humanoid yapıdaki en iyi robottur. Hareket hesaplama, zıplama, koşma, dans etme ve içecek taşıma gibi birçok işlemi yapabiliyordu(Anonim).

2011 yılında Mars'a inceleme amacı ile gönderilen robot Curiosity 2012 yılında Mars'a başarılı bir iniş yaptı. Tekerlekli yapıya sahip olan robotun yüzeyde zorlandığı izlendi. Eklemli yapılı robotlar böylece daha fazla önem kazanmış oldu.



Şekil 1.8. İki bacaklı Humanoid robot ASIMO

Kodomoroid ismi verilen insan görünümlü robot 2014 yılında Japonya'da bir haber kanalında haber sundu(Yamanol,2012).

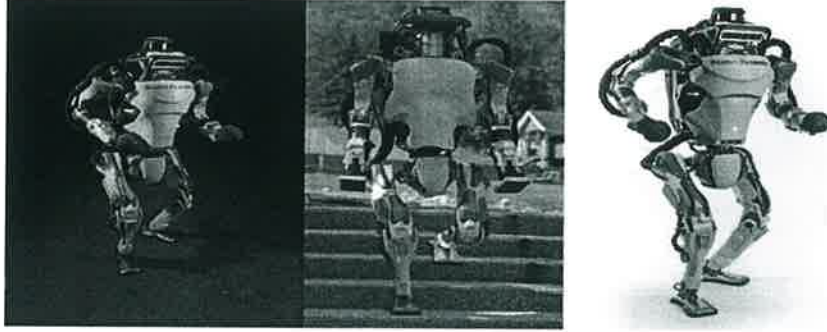
Robotların filmlerde animasyon olarak değil gerçekten oynaması amacı için yapılan çalışmalar sonucunda 2015 yılında Şekil 1.9'da gösterilen Geminoid F isimli insansı robot geliştirilmiştir. Geminoid Fhumanoid RobotSayonora filminde başrol oynayarak tarihe geçmiştir.



Şekil 1.9. İnsan görünümlü robot Geminoid F

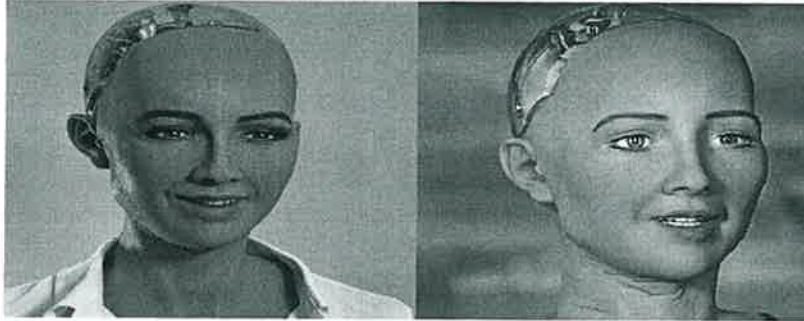
Günümüzde bacaklı robotların gelişimi hızlanmak birlikte Boston Dynamics yaptığı çalışmaları mükemmel bir boyuta taşıyarak 2016 yılında Atlas'ı dünyaya duyurdu(Yamanol,2012).

Şekil 1.10'da gösterilenAtlas Boston Dynamics firmasının üzerinde çalıştığı iki bacaklı robotların son serisidir. Mükemmel bir dengeye sahip olan Atlas'ın kol, bacak ve gövde hareketleri aynı anda kontrol edilmiştir. Atlas'ın hareketini kolaylaştırmak amacı ile tasarımında 3D baskıdan faydalanılmıştır(Anonim).



Şekil 1.10. İki bacaklı robot Atlas

2016 yılında Hanson Robotics tarafından tanıtılan insansı robot Sofya sahip olduğu sinir ağları, makine algısı, konuşma ve doğal dil işleme, sorulan sorulara cevap verebilme özellikleri ilegünümüzdeki en donanımlı humanoid robottur. Sofya'nın bir televizyon programında sunucu ile diyalogu sırasındaki görüntüler Şekil 1.11'de gösterilmiştir.



Şekil 1.11. İnsansı robot Sofya

Geçmişten bu zamana sürekli üzerine çalışmalar yapılan mobil robotların daha ileri seviyelere taşınabilmesine teşvik amacıyla günümüzde birçok etkinlik düzenlenmektedir. RoboCup otonom kontrollü mobil robot yarışması bunlardan bir tanesidir. Bu yarışmaların genel amacı yarışçı konumundaki mobil robotların zorluklar karşısındaki tepkilerini ölçmek istenilen görevleri yapabilme kabiliyetini incelemektir(Willumsen, 2019).

## 2. EKLEMLİ ROBOTLAR

Robotun hareket edebilmesi için hareket mekanizmalarına ihtiyacı vardır. Bu hareket mekanizmaları her düzeyde hareket edemeyebilirler. Her yüzey için farklı hareket mekanizmaları gerekebilir. Örneğin arazi şartlarında bir tekerlekli hareket sistemine sahip robot oldukça zorlanır ve hatta bazı engelleri aşamaz. Bu tür yüzeyler için Şekil 2.1’de olduğu gibi eklemli yapılara ihtiyaç vardır. Aynı zamanda düz bir yüzeyde eklemli yapılı hareket sistemi tekerlekli yapıdaki hareket sistemi kadar hızlı hareket edemez ve görevini yerine getiremeyebilir.



Şekil 2.1. Örnek bir Hexapod yapılı robot

### 2.1. Neden Eklemli Yapı?

Robotlara bacak eklemek tekerlekli hareket mekanizmasına göre daha zor ve karmaşık bir işlemdir. Uzun zamandır insanlar robotları yürütebilmek ve istenilen işlemleri yaptırabilmek için çalışmalar yapmaktadır. Doğaya bakıldığında tekerlekli yapıya herhangi bir örnek yoktur. Tüm canlılar zorlu arazi şartlarına rahatlıkla uyum sağlayabilmek için eklemli bacak yapıları ile var edilmiştir. Örneğin bir örümceğin tekerlekli bir yapı ile tavana tutunma şansı yoktur. Eklem bacaklı yapının tekerlekli yapıya göre bir kaç avantajından bahsetmek gerekirse;

Eklem bacaklı hareket mekanizması, Şekil 2.2’de görüldüğü üzere tekerlekli mekanizmanın hareket edemeyeceği yüzeylerde hareket sağlayabilir. Tekerlekli hareket mekanizmasının engebeli alanlarda hareketi istendiği durumlarda yüzeye göre tekrar tasarlanması gerekir.

Eklem bacaklı yapılar yüzeylerde bulunan engelleri adım boyutunu yükselterek veya zıplayarak aşabilir. Tekerlekli yapıların böyle bir özelliği yoktur. Farklı bir yol izlemek zorunda kalabilir veya hareketine devam edemeyebilirler.



Şekil 2.2. Eklemlı bacak yapısına sahip örnek robot görüntüsü

Tekerlekli yapıların da eklemlı yapılara göre birçok avantajı vardır;

Tekerlekli hareket mekanizmaları eklem bacakları sistemlere göre daha basit mekanik yapıya ve daha kolay hesaplamalara sahiptir. Kararlılık durumu deęişen eklemlı yapılara göre daha kolay tasarlanabilir ve bakımları daha kolay yapılabilir.

Tekerlekli hareket yapıları eklem bacaklı yapılara göre daha ucuzdur.

Düz yüzeylerde tekerlek yapılı hareket eklem bacaklı yapıya göre daha hızlı ve verimlidir(Bapat, 2016).



Şekil 2.3. Tekerlekli bacak yapısına sahip örnek robot görüntüleri

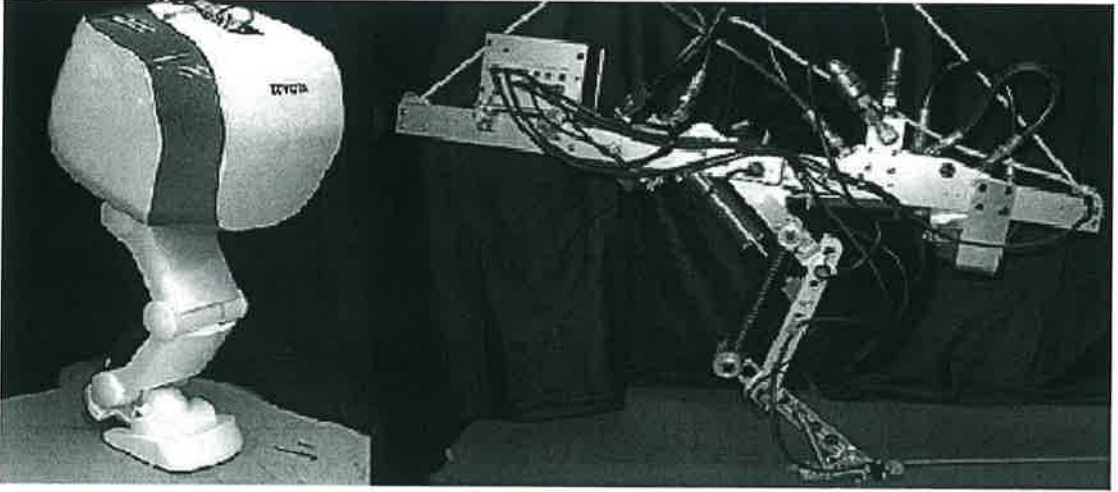
## 2.2. Eklemlı Yapının Çeşitleri ve Gelişimi

Uzun yıllar önce tekerlekli yapının insanoęlu tarafından kullanılmasına başlanmasına rağmen zaman içerisinde doğadan da esinlenerek eklemlı yapılara ihtiyaç duyulmuştur. Eklemlı sistemler tasarlanıp zaman içerisinde geliştirilmiştir. Bu gelişim hala devam etmektedir. Hız, kontrol ve bakım açısından zorlayıcı faktörlere sahip olmasına rağmen yüzey üzerindeki sınırsız hareket yeteneğinden dolayı eklemlı yapılar vazgeçilmez yapılardır.



### 2.2.1. Tek bacaklı robotlar

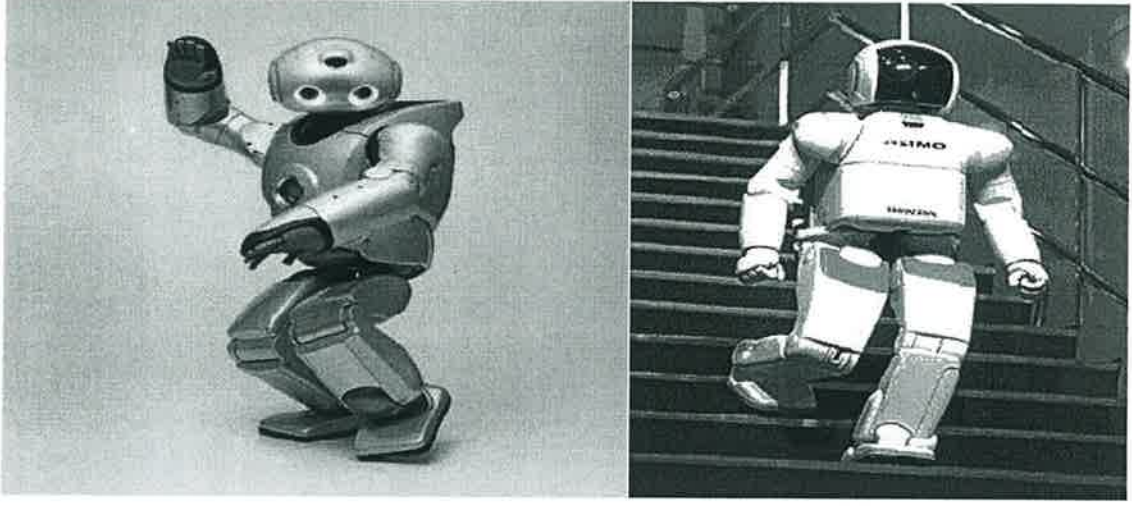
Bir robotun yer çekimine karşı dengede durabilmesi için en az bir bacağına sahip olması gerekir. Bir bacaklı robotlar dengede durabildiği gibi zıplayarak hareket ederler zıplamayı bıraktıkları takdirde dengelerini koruyamazlar. Tek ayak yapısına sahip oldukları için ağırlık merkezleri ayağın tam ortasındadır. Rahatsız edici etkilere karşı ağırlık merkezlerini değiştirerek denge konumunu korumaya çalışırlar. Ayrıca tek bir bacak yapısı olduğu için bacaklar arası koordinasyon ve yürüyüş şekillerine ihtiyaç yoktur. Dönme ve açılma hareketi gereken durumlarda zıplayarak kolaylıkla hareket edebilirler. Bacak sayısı az olduğu içinde enerji açısından verimli robotlardır. Toyota'nın geliştirdiği tek bacaklı robot çalışmaları Şekil 2.4'de gösterilmiştir(Anonim).



Şekil 2.4. Tek bacaklı robotlar

### 2.2.2. İki bacaklı robotlar

Humanoid(insansı) robotlara ilgi duyulmaya başlanması ile birlikte robotlarda iki ayaklı tasarımlar ortaya çıkmıştır. Robotların insanlara benzetilmek istenmesindeki ana etken insanların yapabildiği görevleri robotlara yaptırabilmektir. Robotlar hareketsiz durumda iken statik olarak dengededirler.

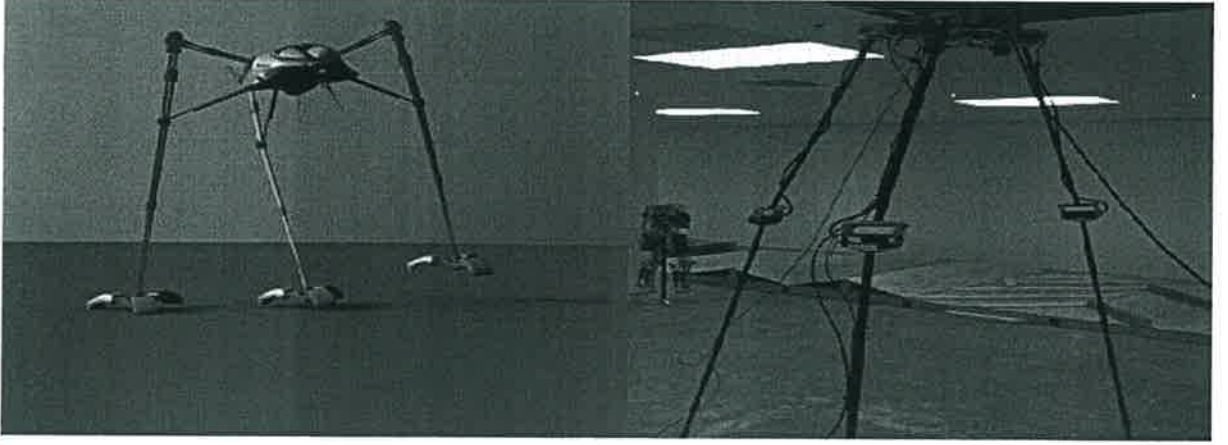


Şekil 2.5. QRIO ve ASIMO humanoid robotlar

Fakat hareket sırasında farklı algoritmalar geliştirmek ve uygulamak gerekir. Şekil 2.5’de gösterilen Honda’nın tanıttığı ASIMO ve Sony’nin geliştirdiği QRIO humanoid robotu yürüme, koşma ve ayakta denge sağlayabilen iki bacaklı robotlara güzel birer örnektir. Bu iki bacaklı humanoid robotlar hareket esnasında bir sonraki adımı denge kurabileceği şekilde hesaplayarak adım atacak şekilde geliştirilmiştir (Anonim).

### 2.2.3. Üç bacaklı robotlar

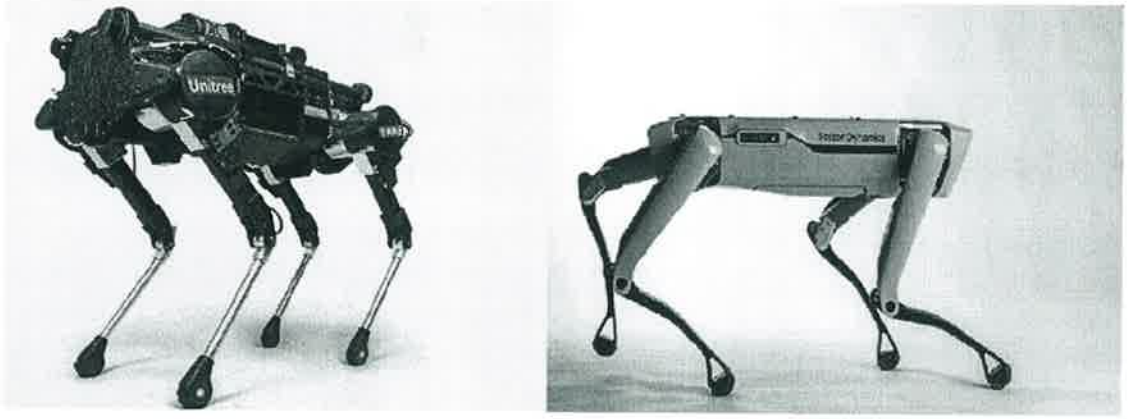
Robotlarda bacak sayısına bağlı olarak denge faktörü değişir. Üç bacaklı robot hareketsiz durumda iken yer ile teması bulunan üç bacak bulunur. Bu durumda robot statik olarak kararlı durumdadır. Hareket sırasında en az bir bacağın hareket etmesi gerektiği için bu durum değişir. Şekil 2.6'da görülen STriDer gibi robotlar üç bacaklı olarak tasarlanmıştır. Üç bacaklı hareketin doğada bilinen bir örneği yoktur. Bu nedenle bu ayak sayısındaki robotlar üzerine diğerlerine oranla daha az bütçeler ayrılmış ve daha az çalışmalar yapılmıştır(Anonim).



Şekil 2.6. Üç bacaklı STriDer robotlar

### 2.2.4. Dört bacaklı robotlar

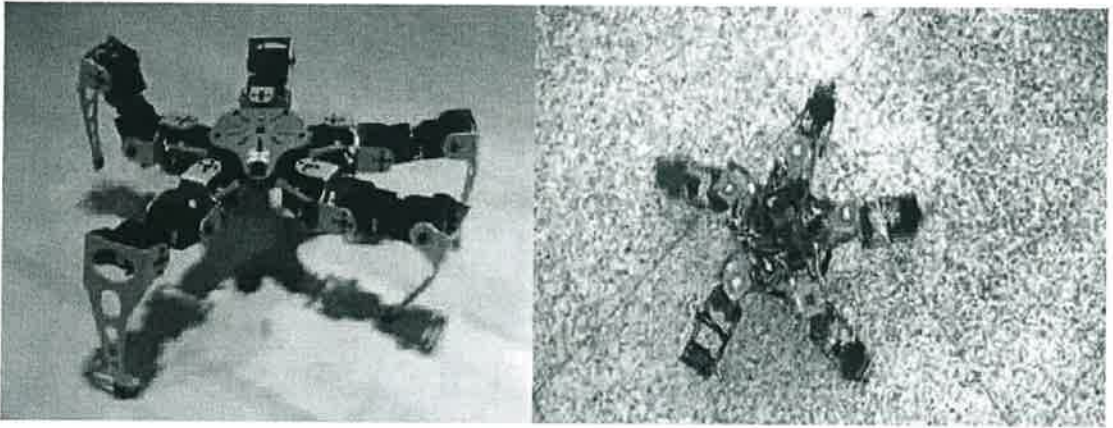
Aslan, kedi, keçi gibi doğadaki birçok hayvan hareket mekanizması incelendiğinde dört bacaklı hareket sistemine sahiptir ve bu hareket mekanizmasını robotlara uygulamak için güzel bir ilham kaynağıdır. Dört bacaklı yürüme sistemi hız ve dengenin aynı anda istendiği durumlarda çok kararlı bir hareket mekanizmasıdır. Yürüyüş biçimi bir çok şekilde dizayn edilebilme imkanı olan bu yürüyüş biçiminin genel olarak kullanımı, bir bacak hareket halinde iken diğer üç bacak statik dengeyi sağlar. Bacaklar sırası ile hareket ederek robotun hareketi sağlanmış olur. Diğer bir yürüyüş biçimi ise bacakları ön ve arka olarak ikiye ayırarak yapılan yürüyüş biçimidir. Bu yürüyüşte kaplanların yürüyüşü temel alınır ve ön bacaklar geriye doğru çekilirken arka bacaklar gerilme için hazırlanır. Hız açısından oldukça verimli olan bu sistemde denge faktörü oldukça düşüktür ve tek bacaklı harekete göre daha az karardır. Boston Dynamics'in yaptığı dört bacaklı mobil robot çalışmalardan bazıları Şekil 2.7'de gösterilmiştir(Anonim).



Şekil 2.7. Dört bacaklı robotlar

### 2.2.5. Beş bacaklı robotlar

Eklem bacaklı robotlar üzerine yapılan çalışmalar günden güne artmaktadır. İlham kaynağı doğadaki canlılar olan bu robotlar robot dünyasındaki en yeni alanlardan birisi olarak görülmektedir. Eklem bacaklı robotlar manevra ve açılı yürüme özellikleri ile tekerlekli hareket yapısının yerini almaya başlamıştır. Tasarımdaki bacak sayısı tasarımcı tarafından robotun kullanılacağı alan göz önünde bulundurularak iyi analiz edilmelidir. Beş bacaklı robotlar Şekil 2.8’de gösterildiği gibi denizyıldızlarının kollarından ilham alınarak tasarlanmıştır. Bu yapı genel olarak 4 bacak dengesi üzerine hareket eder. Beşinci bacağı manevralar ve hızlanmak için kullanan bu tasarım fazla tercih edilmemiştir(Zamri ve ark., 2009).

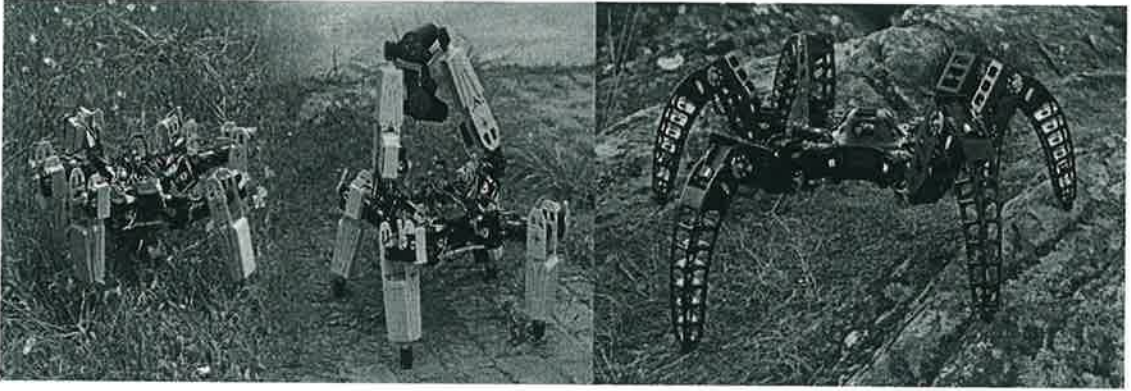


Şekil 2.8. Beş bacaklı denizyıldızı robotlar

### 2.2.6. Altı bacaklı robotlar

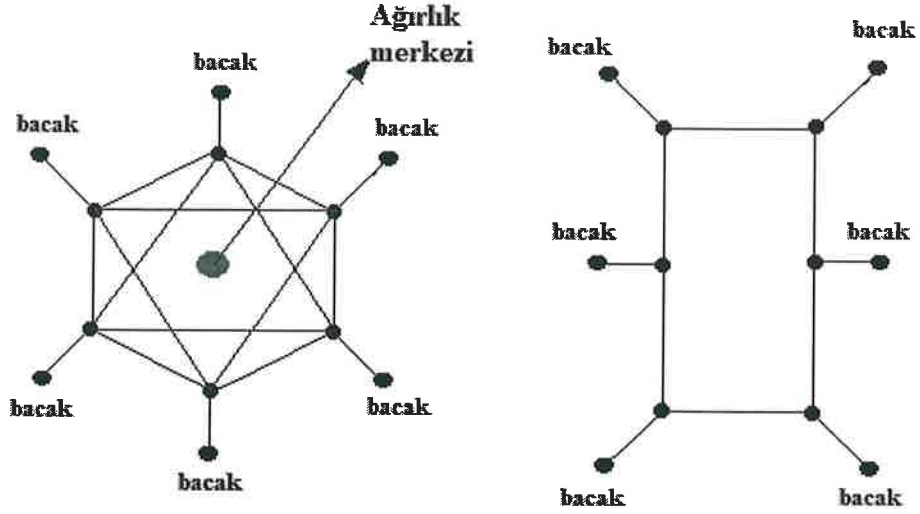
Robotların bacak sayılarının artması kararlılığın artmasını sağlamaktadır. Altı bacaklı robotlar hareketsiz durumda iken ve hareket ederken diğer türlere göre daha kararlıdır. Böceklerden esinlenilerek tasarlanan bu robotların yürüyüş biçimleri de örümcekler

incelenerek oluşturulmuş ve Şekil 2.9’da gösterilen örnek robot görünümleri tasarlanmıştır. Hexapod robotlar için en yaygın olarak kullanılan Wave, Tripod ve Ripple yürüyüş biçimleridir. Wave yürüyüşte bacaklar hareket ettiğinde dalga desenleri meydana gelir. Bacaklar ikili olarak hareket ederler. Diğer dört bacak yerle temas durumunda olduğu için statik kararlılığı yüksek bir yürüyüş biçimidir. Tripod yürüyüşte ise bacaklar üçerli olarak hareket ettirilir. Hareket sırasında kalan üç bacak yer ile temas halindedir. Dalga yürüyüşe göre daha az kararlı olan bu yürüyüş biçimi oldukça hızlı bir yürüyüş biçimidir(Anonim).



Şekil 2.9. Altı bacaklı örümcek robotlar

Altı bacaklı robotların bacaklarının şasi üzerinde yapılandırılması için birçok çalışma yapılmıştır. Çalışmaların çoğu iki çeşit şasi üzerine yoğunlaşmıştır. Bunlar altıgen yapılı şasi ve dikdörtgen yapılı şasi yapılarıdır. Altıgen yapılı şaside bacaklar 60 derece aralıklarla dizilir. Kafa ve kuyruk kısımları için yeterli boşluk mevcut değildir. Dikdörtgen yapılı şaside ise bacaklar sağa ve sola üçerli olacak şekilde dağılır. Kafa ve kuyruklu robot tasarımları için sıkça tercih edilen bir şasi çeşididir. Hexapod robotun bacak yapıları, örümceklerin bacaklarına benzer bir şekilde iki eklemden başlayarak altı eklem kadar çıkan yapılardan oluşturulabilir. Dikdörtgen yapılı bacak dizilimi doğadaki hayvanların yapılarını daha çok andırmaktadır. Çünkü bu bacak diziliminde kuyruk ve bacak için ayrılmış ön ve arka taraflar bulunmaktadır. Hareket sırasında ileri yöne hızlı hareket sağlayan bu yapı dönme ve geriye hareket sırasında zorluklar yaşatmaktadır. Altıgen yapılı tasarım bu konuda daha avantajlıdır. Simetrik bacak dizilimi sebebiyle dönüşlerde ve açılı hareketlerde daha etkili bir hareket yöntemidir(Shahriari, 2013)



Şekil 2.10. Altıgen ve dikdörtgen düzende hexapod yapı

Şekil 2.10'da görülen altıgen yapıyı tasarımda ağırlık merkezi faktörü görülmektedir. Hexapod yürüyüş her ne kadar diğer bacak çeşitlerine göre daha dengeli bir yapıda olsa da ağırlık merkezinin korunması gerekmektedir. Tek eklemlili yürüyüşte bir bacak harekete başladığında diğer bacaklar yere temas ediyor olduğu için ağırlık merkezi kaldırılan bacağı doğru çok az kayar fakat bu kayma denge üçgeni dışına çıkmadığı için ihmal edilebilir. İki bacaklı harekette ise aynı yönde iki bacak kaldırıldığında ağırlık merkezi üçgenden dışarı çıkmaya çok yaklaşır fakat denge korunur ve robot ayakta kalmaya devam eder. Bu iki yürüyüş biçiminde bir sıkıntı olmamış olsa da hareketin devam ettiği yolun eğimine dikkat edilmelidir. Üçayaklı yürüyüş biçiminde aynı yöndeki ayaklar ayağa kaldırıldığı takdirde ağırlık merkezi üçgen dışına taşar ve robot dengesini kaybeder. Bu istenmeyen bir durumdur. Ağırlık merkezi sürekli üçgen içerisinde tutulmak istenir.

### 2.3. Hexapod Robotların Yürüyüş Şekilleri

Doğada bulunan örümceklerin birçok yürüyüş şekli bulunmaktadır. Bu yürüyüş şekilleri canlıların durumunu kolaylaştıracak en uygun hareketleri yapması sonucu oluşmuştur. En yaygın olarak bilinen ve robotlara uygulanan üç yürüyüş şekli mevcuttur. Bunlar Tripod (üçlü) yürüyüş, Wave (dalga) yürüyüş ve Ripple yürüyüşüdür. Yürüyüşler sırasındaki bacak hareket kombinasyonları Tablo 2.1'de gösterilmiştir.

TRIPOD	A	B	A	B	A	B
R1						
R2						
R3						
L1						
L2						
L3						
WAVE	A	B	C	D	E	F
R1						
R2						
R3						
L1						
L2						
L3						
RIPPLE	A	B	C	A	B	C
R1						
R2						
R3						
L1						
L2						
L3						

Tablo 2.1. Hexapod robot için Tripod, Wave ve Ripple yürüyüş

### 2.3.1. Tripod yürüyüş şekli

Bir Hexapod robot şasi yapısına göre sağ ve sol ayaklardan veya ön ayaklar, arka ayaklar ve yan ayaklardan oluşabilir. Fakat ayak sayısı her iki şasi yapısı türünde de aynı olduğu için yürüyüş şekillerindeki bacak hareket sırası aynıdır. Tripod yürüyüş biçiminde Tablo 2.1’de görüldüğü üzere bacaklar A ve B olmak üzere iki periyotta hareket ederler. A periyotunda 2-4-6 numaralı bacaklar ileri yönde hareket eder. Birinci periyot hareket halinde iken ikinci periyottaki bacaklar hareketsiz kalarak robotun dengeli bir biçimde ayakta kalmasını sağlar. A periyodu hareketini tamamladıktan sonra B periyotundaki 1-3-5 numaralı bacaklar ileri yönde hareket eder. B periyodunun hareketi sırasında A periyodu bacakları robotun dengesini sağlar. A ve B periyodları tamamlandıktan sonra robotun hareket döngüsü tamamlanmış olur. Bu periyodlar sürekli tekrarlanarak robotun hareketi sağlanmış olur. Bu yürüyüş biçimi hexapod robot yürüyüş biçimlerinden en hızlı olan yürüyüş biçimidir. Denge kararlılığı diğer yürüyüşlere göre daha az olmasına rağmen üçayak ile denge yeterli görüldüğü için hız istenilen yerlerde sıkça kullanılan bir yürüyüş tipidir.

### 2.3.2. Wave yürüyüş şekli

Dalga yürüyüş olarak dilime çevrilen bu yürüyüş biçimi robotlar için en güvenli ve dengeli yürüyüş biçimidir. Adının dalga olmasının sebebi bacakların sırayla hareket ettiği sırada oluşturduğu simetrik dalga şekilleridir. Robotun tam bir hareket döngüsü altı periyotta tamamlanır. Her periyotta bir bacak hareket sağlar. Hareket başlangıcı A periyodu ile olur ve birinci bacak ileri yönde hareket eder. Birinci bacağın hareketi sırasında diğer beş bacak yer ile temas halindedir ve maksimum denge sağlarlar. Bu nedenle arazi ve eğimli yerler için oldukça ideal bir yürüyüş biçimidir. Bacaklarda hareket A periyodundaki birinci bacak hareketini tamamladıktan sonra B periyodundaki ikinci bacak harekete başlar. Bu esnada A periyodunda olduğu gibi yine beş bacak yer ile temas halinde denge sağlar. Wave yürüyüş biçiminde her zaman beş bacak ile denge sağlanır. Periyotlar birbirini sıra ile takip ederek en son periyod olan F periyoduna geçilir. F periyodu da hareketini tamamladıktan sonra bir hareket döngüsü tamamlanmış olur ve döngü başa döner. Bu yürüyüş biçiminde bacaklar tekli hareket ettiği için oldukça yavaş bir hareket sağlanır. Hızın önemli olmadığı dengenin önemli olduğu zorlu arazi şartlarında tercih edilen bir yürüyüş biçimidir.

### 2.2.3. Ripple yürüyüş şekli

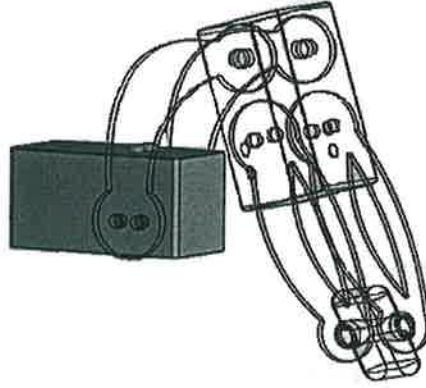
Bu yürüyüş biçimi karmaşık bir yürüyüştür. İki fazdan oluşan bu yürüyüş biçiminde fazlarda oluşan dalga döngüsü üst üste gelmez. Yani bir periyodun karşısındaki periyot tamamen faz dışı olan dalgalar üretir. Bu nedenle bu yürüyüş biçimi oldukça hızlı ve dengelidir(Anonim).

Tablo 2.1'de görüldüğü gibi yürüyüşün gerçekleştiği bacaklar ikili hareket ederler. Üç periyoddan oluşan bu yürüyüş biçiminde hareket sırasında yer ile temas halinde olan dört bacak bulunur. Bu robotun statik denge kararlılığı için yeterli bacak sayısıdır. A periyodun da R1 ve L1 ön bacakları hareket ederler. A periyodun da hareket tamamlandıktan sonra B periyodu bacakları olan R2 ve L2 bacakları hareket ederler. B periyodu da tamamlandıktan sonra son periyod olan C periyoduna geçilir. R3 ve L3 arka bacakları istenilen yönde harekete başlar ve hareket tamamlandığında tam bir hareket döngüsü tamamlanmış olur. Hız ve denge konusunda etkili olan bu yürüyüş biçimi sıkça tercih edilen bir yürüyüş biçimidir.



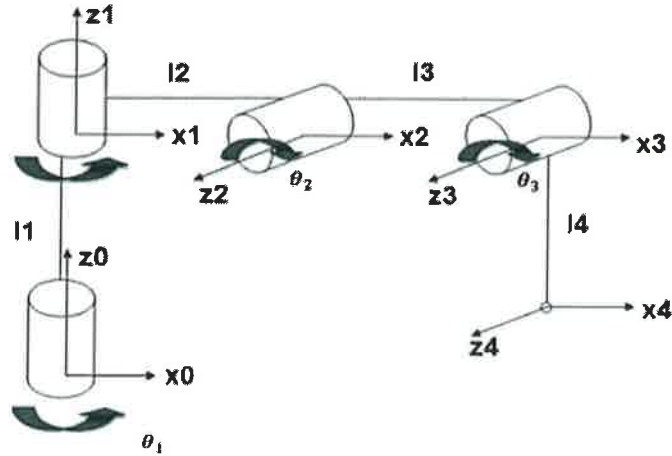
### 3. ROBOT KİNEMATİĞİ

Kinematik denklemlerde hareket başlangıcını sağlayan eklem ile hareketin bittiği uç işleyici arasında bir ilişki kurulur. Kurulan ilişkiden sonra hareketi gerçekleştirecek olan uzuv üzerindeki eklemlere koordinat sistemi tanımlanır. Kinematik denklemler kontrolü yapılacak ayağın veya herhangi bir endüstriyel kolun yapısında bulunan dönel veya prizmatik eklemlerin sayısına göre çıkarılır. Eklemler belirlendikten sonra eklemlerin kinematik denklemleri her bir eklem için çıkartılıp bu denklemler kullanılarak ana denklem çıkarılır.



Şekil 3.1. Hexapod örümcek robotun bacağına CAD görünümü

Robotların eklemlerinin kinematik denklemlerini çıkarmak için birçok yöntem uygulanmaktadır. En yaygın olarak kullanılan yöntem Denavit-Hartenberg yöntemidir. Bu yöntemde D-H tabloları oluşturularak robot ileri yön kinematik denklemleri çıkarılmaktadır. Şekil 3.1'de 3D ortamda tasarlanan robotun bir bacağı gösterilmiştir. Bu bacağın eklem yapısına göre Şekil 3.2'de gösterildiği gibi koordinat sistemi eklenir.



Şekil 3.2. Eklemleri yapı üzerine koordinat sistemi yerleştirilmesi

### 3.1. İleri Yön Kinematığı

İleri yön kinematik denklem çıkarma yöntemlerinden en yaygın olan Denavit-Hartenberg yöntemin de denklem çıkartma esnasında dört temel değişken kullanılır. Değişkenler iki eksen arasındaki uzuv uzunluğunu ifade eden  $d$ , komşu eksenler arasındaki açıyı temsil eden  $\theta$  eksen açısı, eklem kaçıklığı denilen eklem kaymasını ifade eden  $\alpha$  ve birbirini takip eden iki uzuv un arasındaki  $\alpha$  eklem açısıdır. Robot üzerinde bulunan tüm eklemlere koordinat sistemi yerleştirildikten sonra dört temel değişkene göre D-H tablosu oluşturulur. Hareket esnasında uzuv uzunlukları ve eksen açıları değişmezken eklem kayması ve eklem açısı değerleri değişebilir (Moncalı ve Ark., 2016).

$\dot{I}$	$\alpha_{i-1}$ [açı]	$\alpha_{i-1}$ [mm]	$d_i$ [mm]	$\theta_i$ [açı]
1	0	0	$l_1$	$\theta_1$
2	90	$l_2$	0	$\theta_2$
3	0	$l_3$	0	$\theta_3$
4	0	$l_4$	0	0

Tablo 3.1. Denavit-Hartenberg değişken tablosu

Tablo 3.1’de verilen değişkenler kullanılarak robotun bacak yapısında bulunan her bir eklemin homojen dönüşüm matrisleri çıkarılır. Çıkarılan bu matrislerin her biri kendi arasında çarpılır. Çarpım sonucu elde edilen  ${}^0_4T$  homojen dönüşüm matrisi robotun ileri yön kinematik denklemidir. Bu denklem aşağıda verildiği gibi elde edilir (Moncalı ve Ark., 2016).

$${}^0_4T = \begin{bmatrix} c\theta_1 c(\theta_2 + \theta_3) - c\theta_1 s(\theta_2 + \theta_3) & s\theta_1 & c\theta_1 (l_2 c(\theta_2 + \theta_3) + l_3 c\theta_3) & 0 \\ s\theta_1 c(\theta_2 + \theta_3) - s\theta_1 s(\theta_2 + \theta_3) - c\theta_1 & c\theta_1 & s\theta_1 (l_2 + l_4 c(\theta_2 + \theta_3) + l_3 c\theta_2) & 0 \\ s(\theta_2 + \theta_3) & c(\theta_2 + \theta_3) & (l_1 + l_4 s(\theta_2 + \theta_3) + l_3 c\theta_2) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (3.1)$$

Yukarıda verilen denklem genel ileri yön kinematik denklemi olup, açıları sırası ile eklemlerin açıları ve gösterilen uzunluklar ise eklem uzuv uzunluklarını ifade etmektedir.

### 3.2. Ters Yön Kinematığı

D-H tablosundaki değişkenleri kullanarak elde ettiğimiz  ${}^0_4T = {}^0_1T {}^1_2T {}^2_3T {}^3_4T$  homojen dönüşüm matrisindeki eşitliğin her iki tarafını  $[{}^0_1T]^{-1}$  ile çarpılır (Bingül ve ark., 2009).

Çarpım sonucunda aşağıdaki denklemleri elde edilir;

$$p_x \cos \theta_1 + p_y \sin \theta_1 = l_2 + l_4[(\cos \theta_2 \cos \theta_3 - \sin \theta_2 \sin \theta_3) + l_3 \cos \theta_2] \quad (3.2)$$

$$p_y \cos \theta_1 - p_x \sin \theta_1 = 0 \quad (3.3)$$

$$p_y - l_1 = l_4[(\cos \theta_2 \sin \theta_3 + \cos \theta_3 \sin \theta_2) l_3 \sin \theta_2] \quad (3.4)$$

Bu denklemler robotun bacağıının konum vektörleridir. Bu eşitlikleri çözebilmek için aşağıda verilen trigonometrik eşitlikler kullanılır.

$$\theta_1 = \arctan2(-p_y, -p_x) \text{ veya } \theta_1 = \arctan2(p_y, p_x) \quad (3.5)$$

$$l_4 \cos(\theta_2 + \theta_3) + l_3 \cos(\theta_2) = c \quad (3.6)$$

$$l_4 \sin(\theta_2 + \theta_3) + l_3 \sin(\theta_2) = d \quad (3.7)$$

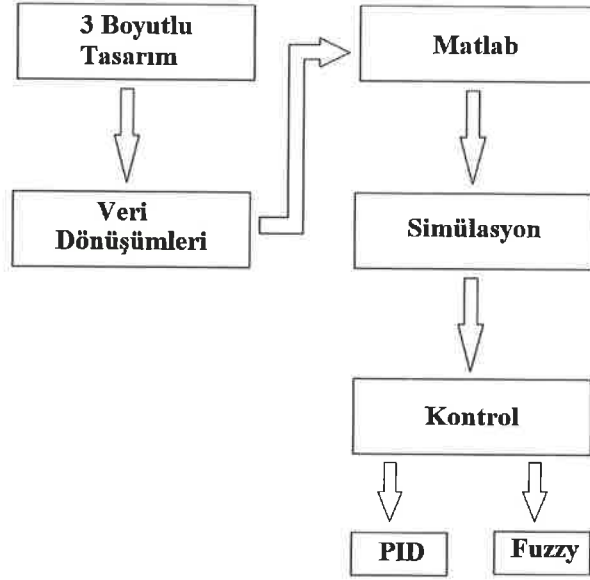
$$\theta_3 = \arctan2(\sin \theta_3, \cos \theta_3) \quad (3.8)$$

$$\theta_2 = \arctan2(rd - sc, rc + sd) \quad (3.9)$$

trigonometrik eşitlikler kullanılır(Moncalı ve Ark., 2016).

#### 4. MATERYAL

Bu tez çalışmasında kullanılan materyaller bu bölümde anlatılmıştır. Tasarım kontrol blok diyagramı Şekil 4.1’de verilmiştir.



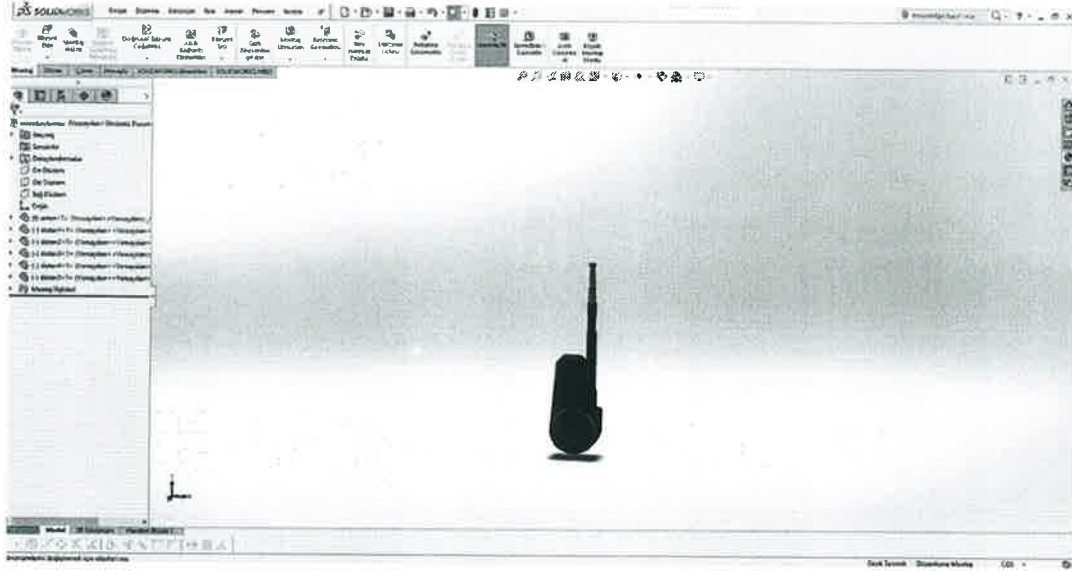
Şekil 4.1. Tasarım ve kontrol blok diyagramı

#### 4.1.Solidworks

SolidWorks Microsoft Windows grafik ara yüzlerini kullanan bir üç boyutlu tasarım programıdır. DassaultSystems tarafından geliştirilen SolidWorks günümüzde mekanik tasarımlarda sıkça kullanılmaktadır. Tasarım aşaması her bir parça için özel açılan parça komutu ve bu parçaların bir araya getirildiği montaj komutlarından oluşur. Bir tasarımın başlangıcında genel olarak iki boyutlu çizim ile başlar. Parçanın ana gövdesi çizildikten sonra çizime komutlar vasıtasıyla üç boyutluluk kazandırılır. Daha sonra parça üzerinde kesme, delme gibi işlemler yapılabilir. Tasarımın her parçası "parça" bölümünde ayrı ayrı tasarlanır. Tasarlanan bu parçalara malzeme kısmından demir, çelik gibi malzemeler atanabilir. Parçalar istenilen renklerde boyanabilir. Parçaları tamamlanan tasarım "montaj" bölümünde birleştirilir. Her bir parçanın birbiri ile temas noktasına ilişki tanımlanır. Bu ilişkiler eş merkezli, birbirine teğet veya çakışık gibi ilişkiler olmaktadır. Montaj aşamasına geçildiğinde parçalar üzerinde düzeltmeler ve yeni parça tasarımları yapmakta mümkündür(Anonim).

SolidWorks tasarım programının piyasada çok yaygın olmasının başlıca sebebi Windows uyumlu ilk CAD programı olmasıdır. Aynı zamanda Autocad gibi birçok iki boyutlu tasarım programı ile uzantı uyumlu çalışması birçok tasarımcının tercih sebebi

olmuştur. İçerdiği eklentiler ile yapılan tasarımın motorlar ile hareketlendirilmesi, yer çekimi eklenmesi ve dayanım analizlerinin yapılması gibi birçok işleme olanak sağlamaktadır.



Şekil 4.2. Solidworks panel anlatımı için örnek görüntü

SolidWorks programına giriş yapıldığında Şekil 4.2’de görüldüğü gibi üst kısımda bulunan menü çubuğunda en çok kullanılan tüm SolidWorks çizim komutları bulunur. Kullanıcının isteğine göre yine menü çubuğundan kısa yol çizim komutları istenilen komutlarla değiştirilebilir. SolidWorks programında çizim esnasındaki o an görülmesi istenmeyen öğeler gizlenebilir ve menüler istenildiği gibi özelleştirilebilir. SolidWorks ekranının sol tarafında bulunan Feature Manager isimli panel, üzerinde çalışılan parçanın çizim esnasında maruz kaldığı tüm işlemleri bize gösterir. Parçayı yeniden düzenlemek, ekleme yapmak veya çıkarmak için bu menü kullanılır. Montaj aşamasına geçildiğinde bu menüden parçalar arasında yapılan montaj ilişkileri de kontrol edilir. Hatalı bir çizim veya montaj ilişkisi uygulandığında panel tarafından hata uyarıları oluşturulur. İçerdiği öğe bilgilerine tıklanarak parçanın müdahale edilmek istenen çizim şekil üzerinde gösterilir (Mamidi, 2012).

#### 4.1.1. Solidworks ile Matlab arasında veri aktarımı

Matlab programlama dilinde görsel, nümeriksel ve kodlama yoluyla programlama gibi farklı programlama türleri vardır. Bu tezde büyük oranda üzerinde çalıştığımız görsel programlama yönteminde solid oluşturarak birçok şekil oluşturulabilmektedir. Fakat bu oluşturulan şekillerin üzerinde oynama yapmak, yapılar eklemek, montajlar gerçekleştirmek olanaksızdır. Bu noktada bir CAD programına ihtiyaç duymak

kaçınılmazdır. İstediğimiz şekilleri tasarım programında ayrıntılı bir şekilde istenildiği gibi tasarlayıp, uzantılar aracılığı ile Matlab ortamında çağırabilmemiz gerekir.

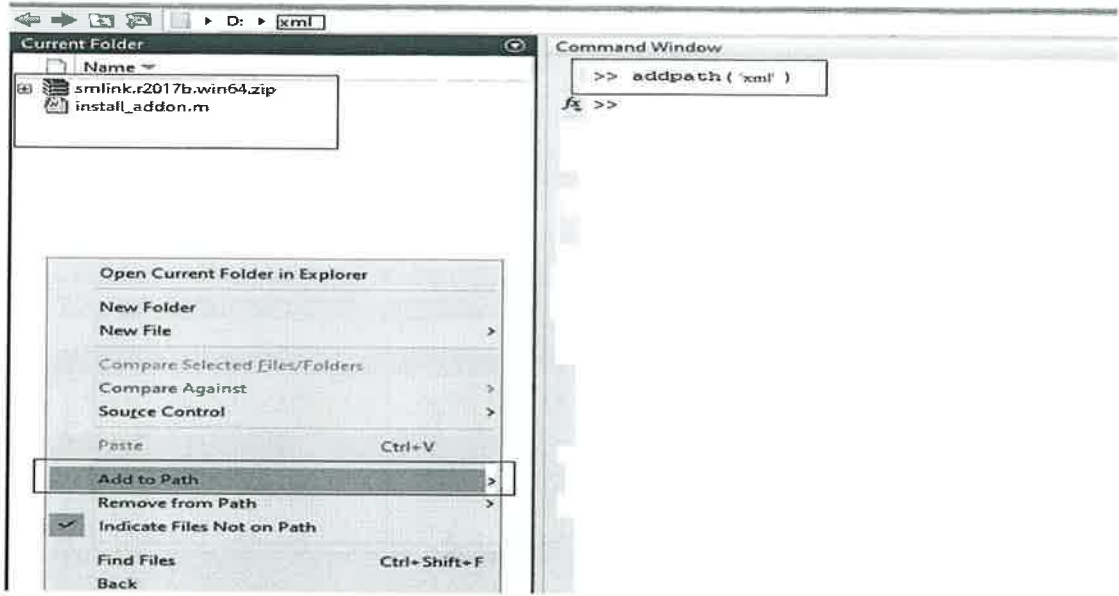
Tezimizde kullanmış olduğumuz üç boyutlu çizim programı olan Solidworks ile kontrol için kullandığımız Matlab arasında bağlantı kurmamız gerekir. Bu bağlantı yolu ile Solidworks programında çizdiğimiz şekillerin veya montajların ağırlık, renk, boyut, derinlik, kalınlık, yoğunluk gibi değerlerini Matlab ortamına aktarabiliriz. Böylece, Matlab ortamında solid oluşturmadan, tasarım programında oluşturduğumuz şekilleri Matlab de kullanabiliriz. Bu işlemleri gerçekleştirebilmemiz için bazı eklentiler indirilip Matlab içerisine kurulması ve bazı aşamalar gerçekleştirilmelidir.

Simscape Multibody Link downloadpage sayfasına gidilerek kullanılan Matlab versiyonuna uygun olan dosya indirilir. Çalışma boyunca Matlab r2017b sürümünü kullanıldığı için Şekil 4.3’de görülen dosyalar indirilmiştir.



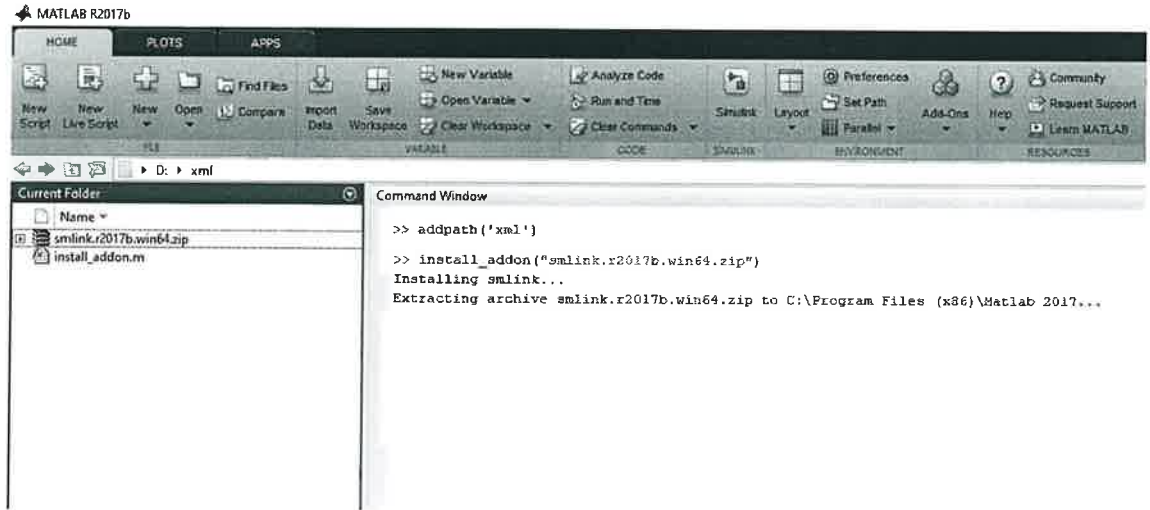
Şekil 4.3. Solidworks ile Matlab ilişkilendirme dosyaları

Eklenti dosyaları indirildikten sonra Matlab yönetici olarak çalıştırılır. Matlab açıldıktan sonra indirilen dosya klasörünün içine giriş yapılır. Açılan bu klasöre addpath komutu ile yol eklenir. Klasör içerisine sağ tık yapılarak da yol ekleme işlemi gerçekleştirilebilir. Şekil 4.4’de işlem ayrıntılı olarak gösterilmiştir.



Şekil 4.4. Eklenti dosyasına yol ekleme

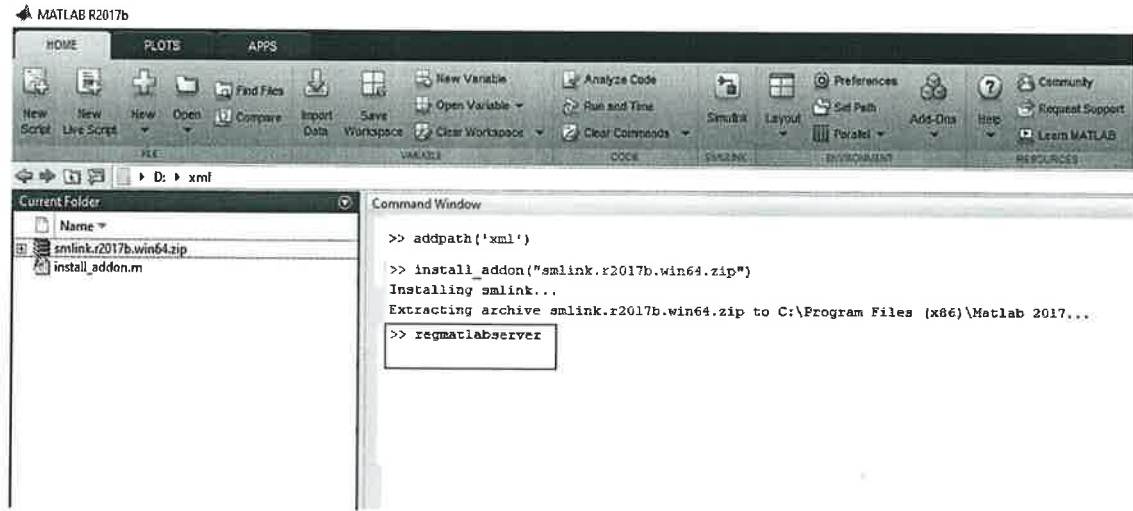
Yol ekleme işlemi tamamlandıktan sonra indirilen zip uzantılı winrar dosyası Matlab'e yüklenir. Bu işlem yapılırken Matlab komut dosyaları içerisinde bu eklentiye görmemiz mümkündür. Yükleme işlemi için komut sayfasına "install\_addon" komutu girilir. Komut girildikten sonra parantez içerisinde eklentinin indirildiği dosya adı girilmesi gerekir. Eklentinin winrardan çıkarılmaması gerekir ayrıca uzantıda isim girilirken yazılmalıdır. Söz konusu işlem Şekil 4.5'de gösterilmiştir.



Şekil 4.5. Eklenti dosyasını Matlab içerisine yükleme

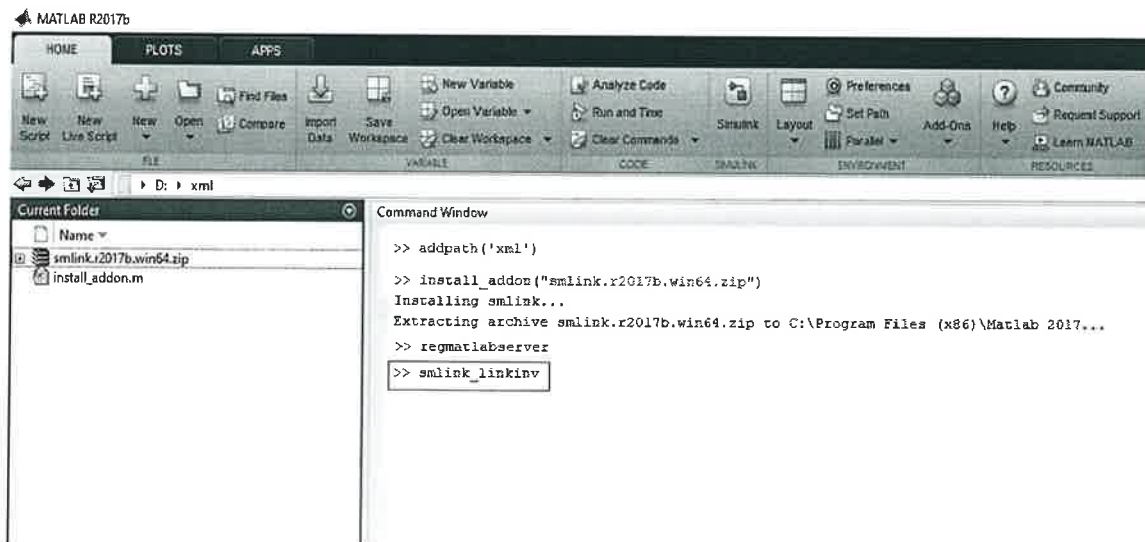
Solidworks, tasarlanmış olan parçaların birleştirilerek oluşturulan montajı, sldprt uzantılı dosyalar olarak her dışa aktardığında SimscapeMultibody Link eklentisi

Matlab'e bağlanmaya çalışır. Bu bağlantının gerçekleştirilmesi için Matlab bir otomasyon sunucusu olarak kaydedilmelidir. Komut sayfasına "regmatlabserver" komutu veya "regserver" komutu girilir. İşlem Şekil 4.6'da gösterilmektedir.



Şekil 4.6. Matlab'in otomasyon sunucusu olarak kayıt edilmesi

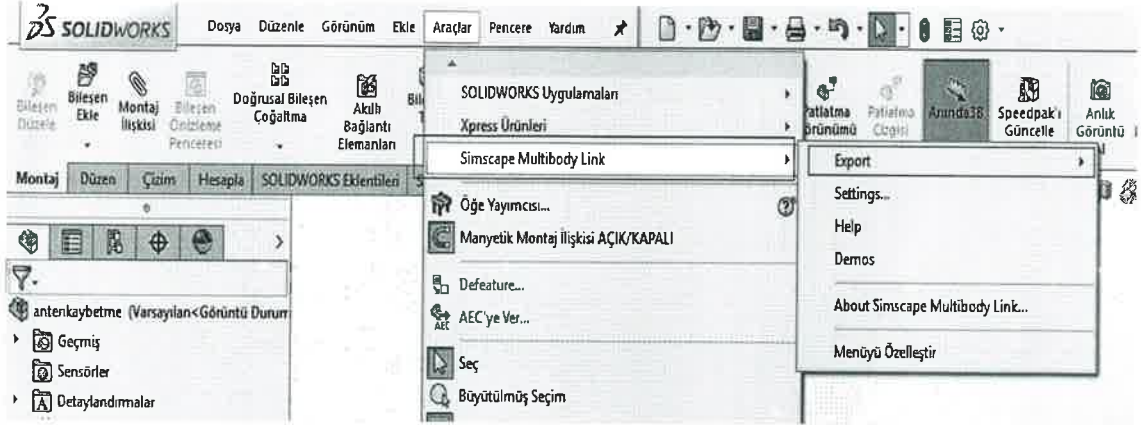
Eklenti yüklendikten sonra AutodeskInventor uygulamasının etkinleştirilmesi gerekir. Bu uygulamayı etkinleştirmek için Matlab komut sayfasına "smlink\_linkinv" komutu yazılır. Bu sayede herhangi bir Cad programına giriş yapıldığı durumda, programın menüsünde SimscapeMultibody Link menüsü seçeneği görülür. Bu seçenek içerisinde bulunan "export" komutu kullanılarak açılan montajda bulunan parçaların her birinin sldprt uzantılı dosyalar ile dışa aktarımı sağlanır. İşlemler Şekil4.7 ve Şekil4.8'de gösterilmiştir.



Şekil 4.7. AutodeskInventor uygulamasının etkinleştirilmesi



Bu adımlar tamamlandığında tasarım programı ortamında ki işlemler tamamlanmış olur. Bu aşamadan sonra Matlab ortamında oluşturulan dosyalar çağırılır. Çağırılan dosyalardaki parça bilgileri Matlab tarafından oluşturulur. Parçalar gerçek boyutlu olarak görsel programlama ortamında elde edilir. Bu aşama tamamlandıktan sonra kontrol aşamasına geçilir.



Şekil 4.8. Solidworks ortamında montajın sldprt uzantılı kaydedilmesi

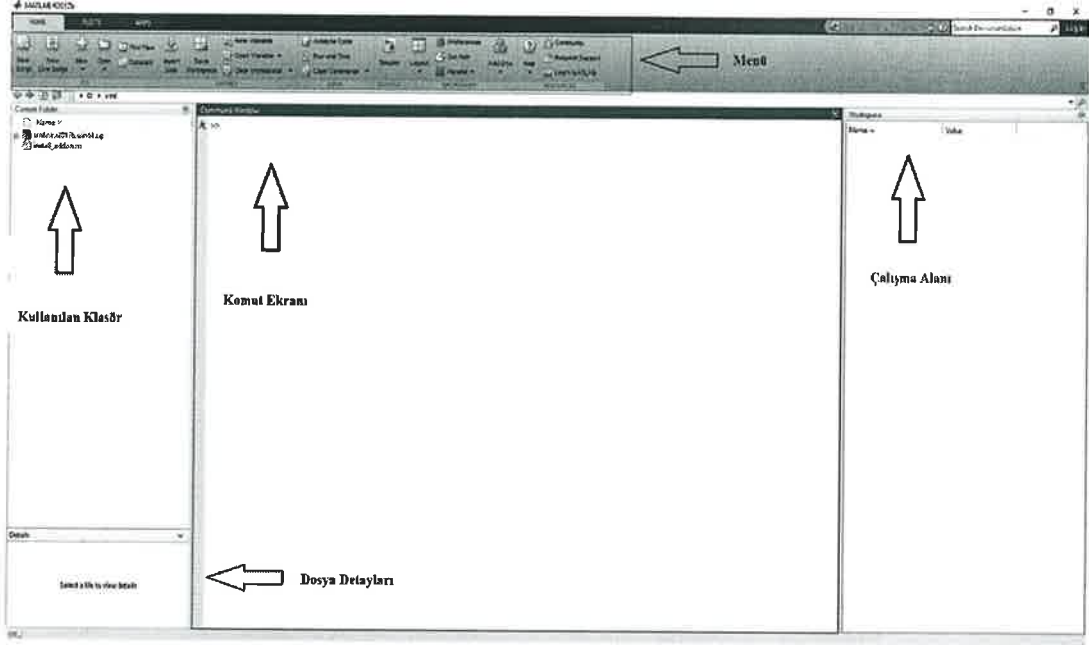
## 4.2. MATLAB

Matlab arka planında sayısal hesaplamalar yapan birçok formülü çözücü olarak kullanabilen bir programlama dilidir. Programlama dilleri segmentinde dördüncü nesil programlama dilleri içinde bulunmaktadır. Bunu daha açık bir şekilde anlatacak olursak, kullanım kolaylığı açısından içerisinde hazır şablonlar, kütüphaneler ve komutlar bulunduran bir programlama dilidir.

Matlab matematiksel programlama dili, MathWorks şirketi tarafından geliştirilmektedir. İçerisinde birçok hazır komut bulunduran Matlab, kullanıcıları için C, C++ ve Java gibi farklı dillerle oluşturulmuş programlar ile çalışmaya da olanak sağlamaktadır. Kontrol mühendisliği tarafından yoğun talep gören Matlab günümüzde eğitim alanında ve mühendislik alanında yaygın olarak kullanılmaktadır. Görüntü işleme alanında sağladığı olanaklar ile eğitim alanında vazgeçilmez olan Matlab analiz ve cebir öğretiminde de kullanılmaktadır (Anonim).

Matlab'e giriş yapıldığında Şekil 4.9'da görüldüğü gibi sol tarafta kullanılan projelerin bulunduğu klasör bulunur. Bu klasörden çalışılacak proje seçilir. Bir proje açılmak istendiğinde Matlab bu klasör içerisinde arama yapar. Boş bir sayfa olarak karşımıza gelen kısım komut ekranıdır. Komut ekranı Matlab ile kullanıcı arasında iletişim sağlar. Basit bir şekilde bu ekranda toplama, çıkarma işlemleri yapılabilir. Örnek olarak Fuzzy kontrol çalışılmak istendiğinde komut ekranına Fuzzy yazıldığında Matlab kullanıcıyı

Fuzzy kontrol ekranı ile karşılaştırır. Yapılan işlemler silinmek istendiğinde silme komutu ile sayfa temizlenebilir(Anonim).

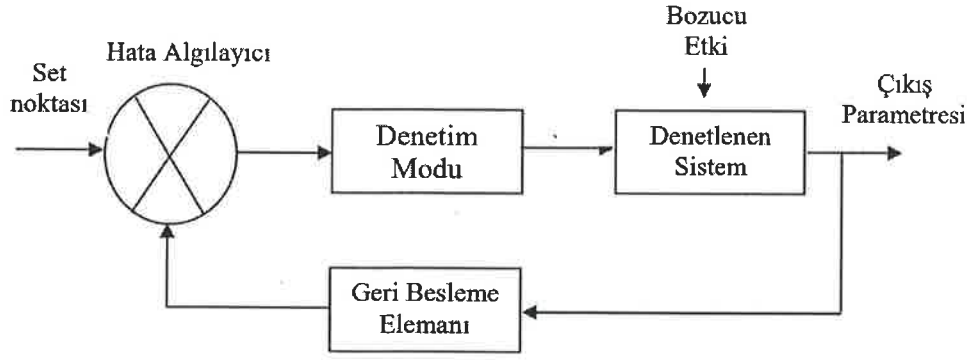


Şekil 4.9. Matlabpanel gösterimi

Matlab ortamında çalışırken oluşturulan sabit değerler, değişkenler sağ tarafta bulunan çalışma alanında saklanır. Bu değer ve değişkenler programlama sırasında kullanıldığında Matlab çalışma alanından bu değişkenlerin değerlerini çağırır. Bu pencerenin amacı programlamaya kolaylaştırmaktır. Ayrıca kayıt edilmesi gerekir aksi takdirde tüm veriler silinir. Kayıt edilen proje açıldığında çalışma alanının da çağırılması gerekir. Matlab programlama dilinde hazır programlar mevcuttur. Bu programlar Matlab fonksiyonlarıdır ve matematikte kullanılan  $y=f(x)$  fonksiyonu ile kullanımları benzerdir(Anonim).

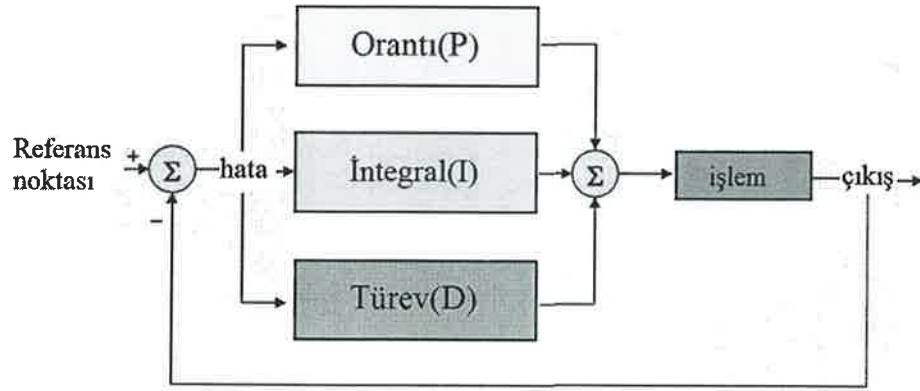
### 4.3. PID Kontrolör

PID kontrol yöntemi robotik alanında yapılan çalışmalarda ve otomasyon sistemlerinde sıkça görülen bir kontrol yöntemidir. Bu yöntem sistemden sürekli geri besleme olarak belirlenen süre içerisinde referans noktası ile kontrol edilen değer arasındaki hata değerini hesaplar ve en aza indirecek çıkış değerleri üretir. Şekil 4.10'da sisteme girişi gösterilen set noktası değeri, denetim modu ve denetlenen sistem parametrelerinden geçerek çıkışa ulaşır. Çıkışa ulaşan bu değer geri besleme elemanı ile tekrar hata algılayıcıya yönlendirilir ve bu değer tekrar sisteme giriş yapar. Bu yol ile hata değeri en aza indirilmek amaçlanır. Şematize edilen döngü PID kontrolörün temel mantığını barındırır.



Şekil 4.10. PID kontrolör temel şeması

PID kontrolör içerisinde barındırdığı oransal(P), İntegral(I) ve Türev(D) terimleri sırası ile hatayı küçültme, hata alanını hesaplama ve zaman hesaplama işlemlerini yapar.P,I ve D parametrelerinin şema üzerinde yerleşimi Şekil 4.11’da gösterilmiştir.



Şekil 4.11. P-I-D kontrolör blok diyagramı

PID kontrolörün içerisinde barındırdığı Orantı, Türev ve İntegral değerlerinin sırasıyla içerisinde barındırdığı  $K_p$ ,  $K_i$  ve  $K_d$  değerleri bu parametrelerin kazançlarıdır. Bu değerler ile sistemin kararlılığı ayarlanır. Kazançlar kullanılarak parametrelerin hesaplanması;

$$P = K_p e(t) \quad (4.1)$$

$$I = K_i \int_0^t e(t) dt \quad (4.2)$$

$$D = K_d e(t) \frac{de(t)}{dt} \quad (4.3)$$

olarak bulunur. Bir fonksiyonun İntegral değerinin Laplace dönüşümü, İntegral değerini “1/s” ile çarparak elde edilir. Fonksiyonun Türev değeri ise fonksiyonu “s” ile çarparak

elde edilir. Parametrelerde elde edilen  $K_p$ ,  $K_d$  ve  $K_i$  kazançlarının “s” domeninde gösterimi:

$$U(s) = \left( K_p + \frac{K_I}{s} + K_D \cdot s \right) E(s) \quad (4.4)$$

$$C(s) = \frac{U(s)}{E(s)} = K_p + \frac{K_I}{s} + K_D \cdot s \quad (4.5)$$

olarak PID kontrolör transfer fonksiyonları elde edilir.

P,I ve D parametrelerinden oluşan PID kontrolörler tam dereceli kontrolörler olarak bilinmektedir. Bu üç parametreye ek olarak iki parametre daha eklenerek kullanıcıya beş parametre kullanma imkanı sağlayan kesir dereceli PID kontrolörler mevcuttur. Kesir dereceli kontrolörlerin temelinde L'Hospital kuralı mevcuttur(Caponetto ve ark.,2002).

#### 4.3.1. Proportional(P)

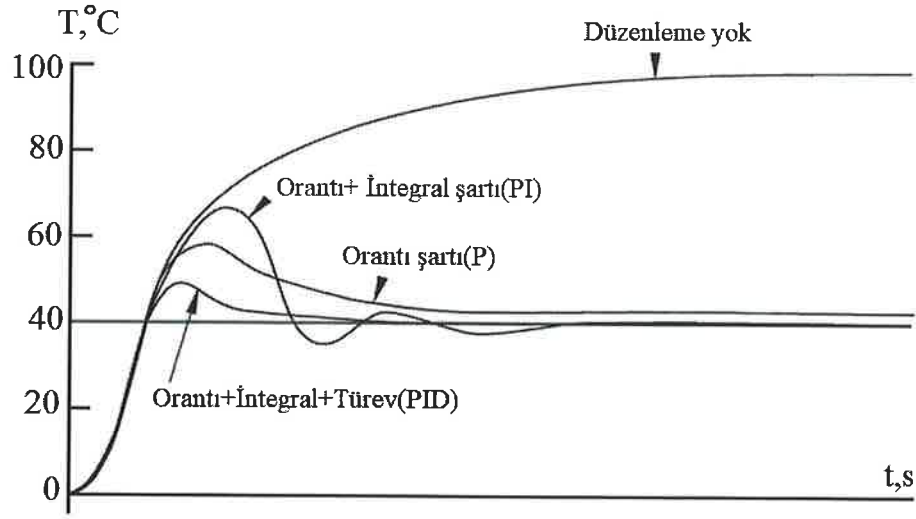
Türkçede orantılı olarak karşılık bulan bu terim sistemdeki hata değerini  $K_p$  adını verdiğimiz bir kat sayı ile çarparak hatayı minimize etmeye çalışır. Buradaki K değeri sistemin orantı kazancıdır. En uygun  $K_p$  değerini seçmenin farklı yöntemleri olduğu gibi deneme yanılma yolu ile de seçilebilir.

#### 4.3.2. Integral (I)

PID kontrolör içerisindeki üç parametreden birisi olan bu terim hata süresini kontrol eder. Aslında tam olarak hata süresi demek yanlıştır. Belirli bir t süresi içerisinde meydana gelen her bir hatanın hata süresine bölümü ile elde edilir. Elde edilen bölüm toplamları kendi aralarında toplanarak değer elde edilir.

#### 4.3.3. Derivative(D)

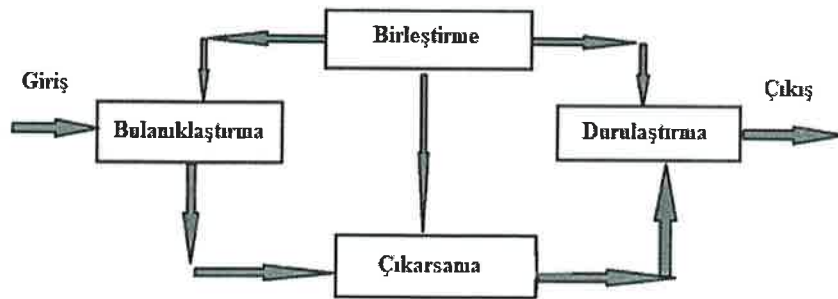
Dilimize türev olarak giren bu terim belirli bir sürede hatada meydana gelen değişimdir. Hataların farkı geçen süreye bölünerek değer elde edilebilir. Genellikle hareket kontrollerinde daha sık kullanılırlar. Sistemi hassaslaştırıp ve hızlandırdıkları için gürültülerden çok etkilenirler bu nedenle süreç kontrol tiplerinde tercih edilmedikleri gibi düzgün sonuçlar vermezler. Bir sinyalin kontrolü üzerine yapılan testte, takip sinyalinin kontrolünde P, I ve D parametrelerinin sırası ile eklenerek sinyale etkileri Şekil 4.12’de gösterilmiştir.



Şekil 4.12. Integral Derivative Proportional değerlerinin sinyale katkı izlenimleri

#### 4.4. Fuzzy Kontrolör

Heisenberg'in ileriye sürdüğü belirsizlik ilkesi 1920 ve 1930 yılları arasında birçok bulanık sistemler üzerine çalışmaların yoğunlaşmasına sebep olmuştur. Bu yoğun çalışmalar neticesinde değer kümesi doğru ve yanlış olmak üzere iki değerden oluşan sistemlere üçüncü bir değer olan belirlenemezlik değeri eklenmiştir. Bulanık sistemlerinin sınırları doğru, yanlış ve belirlenemezlik değerleri ile belirlendi. Heisenberg'in belirlenemezlik ilkesi olarak bilinen ilke bilimi çok değerli belirsizliğe yöneltti. Belirlenemezlik ilkesini 1930'ların başında çok az araştırmacının benimsemesine rağmen, ilkeyi benimsemiş olan Lukasiewicz, Gödel ve Black yaptıkları çalışmalar sonucunda bulanık sistemleri geliştirdiler (Küçüköncü, 2007).



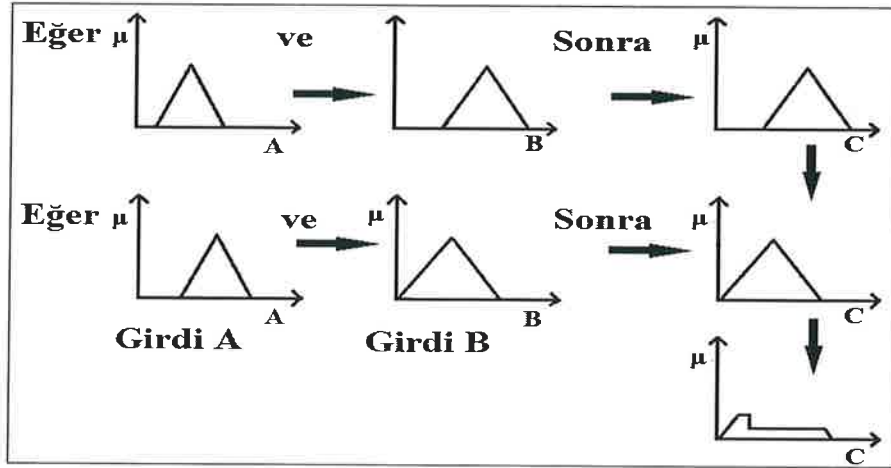
Şekil 4.13. Bulanık sistem mimarisi

Bulanık sisteme gelen girdilerin tamamı ile çıktılarının tamamını birbiri ile ilişkilendirir. Bu ilişkilendirme sonucunda belirlenen aralıktaki kümelerin ve uyulması gereken kuralların tanımlamasını yapar. Bu sebepten dolayı bulanık sistemlerin çalışma mantığı

matematikte bulunan neden sonuç ilişkisine çok benzer bir çalışma mantığıdır. Bulanık sistemlerin içerdiği önemli faktörlerden birisi olan bulanık kurallar, kontrolör ile gerçekleştirilmek istenilen amacı gerçekleştirmeye uygun kullanıcı tarafından belirlenen bulanık kurallardır. Bulanık sistem mimarisini bulanıklaştırma, çıkarsama, birleştirme ve durulaştırmadan oluşan temel dört parametre oluşturur. Bu parametrelerin işlev sırası Şekil 4.13'deki şemada gösterilmiştir. Bulanık mantık sistemleri genel sistemlerin bir biçimi olarak kabul edilir. Bulanık sistemlerde en yaygın olarak kullanılan sistem, eğer ve ise parametleri üzerine kurulan kural tabanlı bulanık mantık sistemleridir(Anonim).

1961 yılında yayınlanan Lütfi Askerzade'nin makalesinden sonra bulanık mantık kavramı ortaya çıkmıştır. Bulanık mantık kavramını oluşturan iki temel bulunmaktadı. Bu temeller bulanık kümeler ve alt kümelerdir. Matematiksel olarak düşünüldüğünde kümenin elemanı var ise değer 1, elemanı olmadığında ise değeri 0'dır. Bulanık mantık yapısında ise kümenin değer 0 ve 1 aralığında bir değerdir ve değer bu aralıkta sonsuz değer alabilir. Matematiksel kümelerdeki az-çok gibi zıt kavramlar bulanık mantığa gelindiğinde biraz az, biraz çok gibi kavramlara dönüştürülerek gerçeğe yakınlaştırılır. Bunun yapılmasındaki neden problemlere en yakın değişkenleri elde edebilmektir(Anonim).

Bulanık mantık sistemlerinin günlük hayatımıza geçişinin öncü uygulamaları çimento sanayi ve kirli su arıtma çalışmalarıdır. Bu alanlarda başarı elde edildikten sonra yayılan bulanık mantık daha sonra asansör ve vinç denetimi gibi bir yerlerde de kullanılmıştır. 1988 yılında Japonya'da bir firma tarafından yeraltı treninde uygulanan bulanık mantık sistemi çok başarılı olmuştur. Günümüzde ev birçok alanda bulanık mantık sistemleri kullanılmaktadır(Küçüköncü,2007).



Şekil 4.14. Bulanık mantık mimarisini

## 5. YÖNTEM

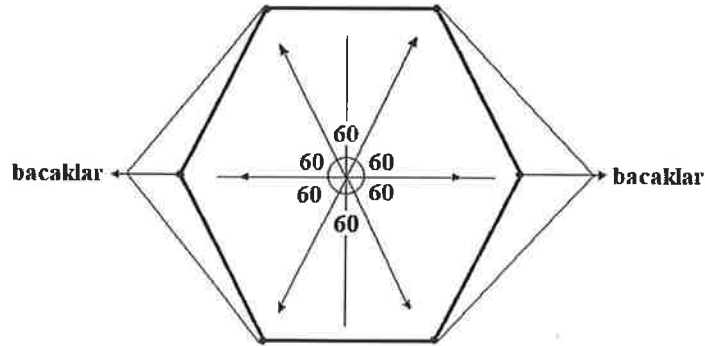
Bu bölümde tezin yapımı sırasında izlenen yollar ve uygulanan adımlar anlatılmıştır. Şekil 5.0'de yöntem şeması verilmiştir.



Şekil 5.0. Yöntem şeması

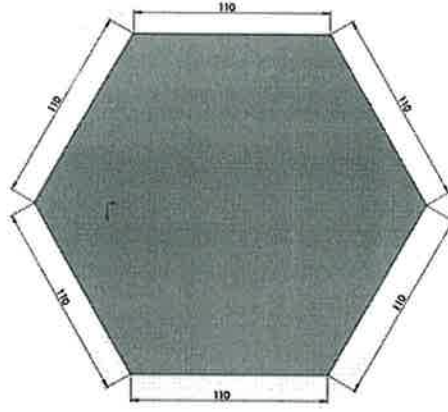
### 5.1. Tasarım Aşaması

Hexapod robotun tasarım aşamasında öncelikle şasi tasarımı yapıldı. Literatür taramasında bahsedildiği gibi iki tür şasi yapısı vardır. Bunlardan biri altıgen yapılı şasi diğeri ise dikdörtgen yapılı şasidir. Üzerinde çalışmış olduğumuz robotun ağırlık merkezinde uzayan boyun yapılı bir kamera bulunacağı için denge faktörü ön plandadır. Bu nedenle altıgen şasi yapısı üzerinde tasarım yapılmıştır.

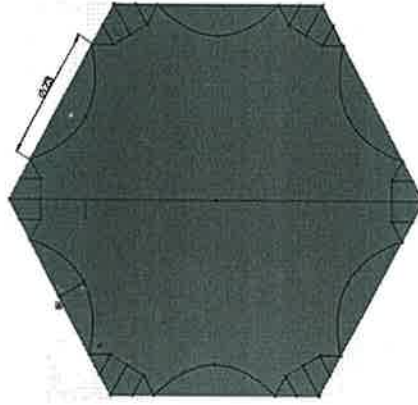


Şekil 5.1. Şasi ön çizim

Şekil 5.1'de görülen ön çizimde bacakların 60 derece aralıklarla şasi üzerine yerleştirildiği görülmektedir. Ön çizime bağlı olarak yapılan bir kenarı 110mm olarak çizilen CAD taslak şasi çizimi Şekil 5.2'de gösterilmiştir.

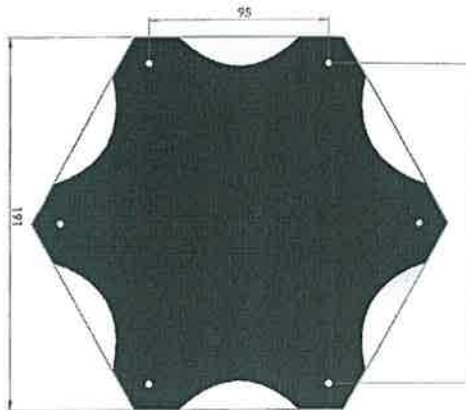


Şekil 5.2. Şasi Cadtaşlak çizimi



Şekil 5.3. Bacak hareket aralığı

Şasinin çiziminde robotun bacakları arasında çapı 73mm olan bir dairenin 15mmlik derinliği kullanılarak her bacağın sağına ve soluna çizim işlemi yapılmıştır. İşlem Şekil 5.3’de gösterilmiştir. İşlemin amacı bacakların hareket sırasında şasiye temas etmemesidir.



Şekil 5.4. Şasi Cad çizimi





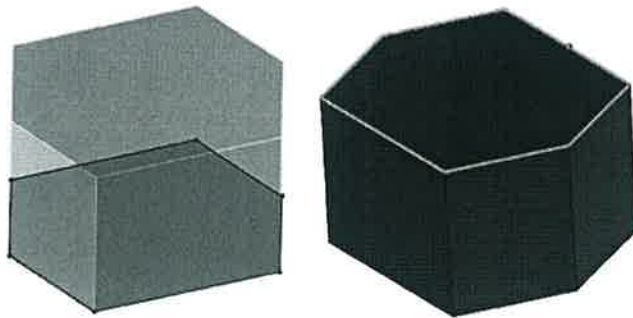
Şekil 5.5. Güç kutusu çizimi

Şekil 5.3’de yapılan çizimlere ekstrüzyon işlemi uygulanması sonucu oluşan çizim Şekil 5.4’de gösterilmiştir. Ayrıca motorların mil girişleri her bacağıın merkez bölgesine açılmıştır. İlk servo motor ve şasi arasındaki bağlantı motorun mili ve şasinin birbirine montajı şeklinde tasarlanmıştır. Şasi üzerine güç kutusu montajı yapılacağı için 2mm derinlikte bir kenarı 94.5mm olan bir altıgen yapı şasinin tam ortasına uygulanmıştır.



Şekil 5.6. Şasinin son hali

Şekil 5.5’de gösterilen çizim işlemine kesme işlemi uygulandıktan sonra oluşan Şekil 5.6’da gösterilmiştir. Bu şekil montajda kullanılan şasinin görünümüdür.



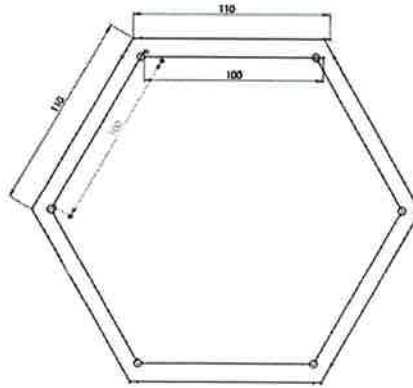
Şekil 5.7. Güç kutusu taslak ve ekstrüzyon

Robotun kontrolünü sağlayacak olan denetleyicinin, sürücünün ve güç sağlayan bataryaların içerisinde bulunacağı güç kutusunun tasarımı 94mm kenar uzunluklu altıgen yapıda yapılmıştır. 2mm et kalınlığı verilerek ekstrüzyon yapılmıştır. Üç boyut kazandırılan güç kutusunun yüksekliği 80mm olarak ayarlanmıştır. Yapılan işlemler Şekil 5.7’de gösterilmiştir.



Şekil 5.8. Güç kutusu

Katı modellemesi tamamlanan güç kutusunun her bir kenarına 50mm radyus verilerek içe doğru boğumlanması sağlanmıştır. Güç kutusunun giydirmeden dışarıda kalacak kısımları kameraya doğru yatay bir şekilde tasarlanmıştır. Bu tasarıma gidilmesinin amacı güneş paneli gibi ek etmenlere olanak sağlamaktır. Katı modelleme yapıldıktan sonra güç kutusunun tam merkezine 25x9mm kenar uzunluklu bir dikdörtgen çizilerek kesim işlemi yapılmıştır. Daha sonra 2mm et kalınlığı bırakılarak şeklin içi boşaltılmıştır. Tasarım Şekil 5.8’de gösterilmiştir.



Şekil 5.9. Şasi giydirme taslak çizim

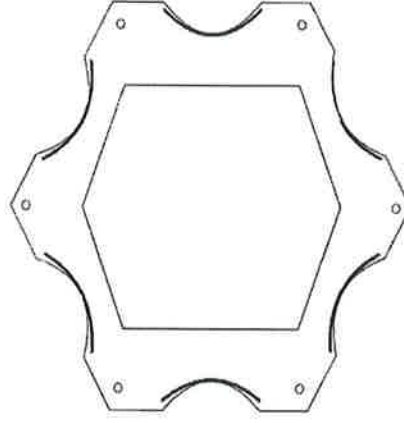


Şekil 5.10. Güç kutusu bölmesi

Şasinin ve servo motorların üzerini örtecek olan şasi giydirmesinin taslak çizimi Şekil 5.9'da gösterilmiştir. Şaside olduğu gibi dış altıgenin kenar uzunlukları 110 mm olarak girilmiştir. Pinlerin dizilimi için çizilen altıgenin kenar uzunluğu ise 100mm olarak girilmiştir. Taslak çizimin katı model haline getirilmesi işleminde 5mm kalınlık sağlanmıştır. Oluşan şekil içerisine bir kenar uzunluğu 94.5mm olan bir altıgen çizilmiştir. Şekil 5.10'da gösterilen bu altıgenin çizilme amacı ise güç kutusunun içerisinden geçecek olmasıdır.

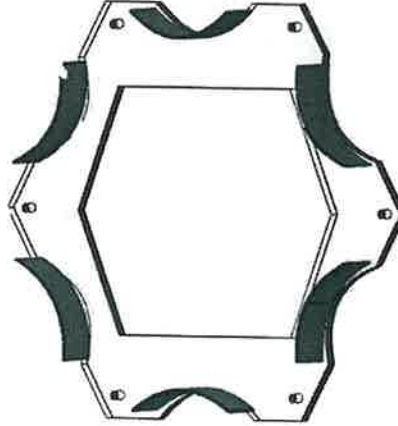


Şekil 5.11. Güç kutusu bölmesi yapımı



Şekil 5.12. Yan kanatçık tasarımı

Şekil 5.10’da gösterilen çizime kesme işlemi uygulandıktan sonra ki oluşan Şekil 5.11’de gösterilmiştir. Oluşan bu şekilde her bacağın kendisi için açılan koruma bölmesine yine 2mm derinliğinde bu sefer çapı 45mm olan bir çember çizilmiştir. Bu çembere 1mm kalınlık verilmiş ve 35mm yükseltme işlemi uygulanmıştır. Servoların ve bağlantı kablolarının görünmesini önlemek için 20mm yüksekliğinde 60mm genişliğinde etek amacıyla kanatçıklar tasarlanmıştır. Tasarım Şekil 5.12’de gösterilmiştir.



Şekil 5.13. Şasi giydirmenin son hali

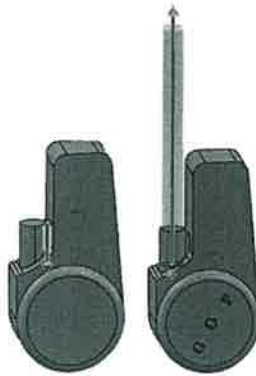
Yapılan işlemler sonunda şasi giydirme Şekil 5.13’de görüldüğü gibi elde edilmiştir. Montajda kanatçıkların şasi ile herhangi bir bağlantısı yoktur. Giydirme güç kutusuna monte edilmiştir.



Şekil 5.14. Anten taslak çizimi

Robotun şasisinin merkezine yerleştirilecek olan kameranın boyun görevini görecek olan anten yapısı tasarımı yapılmıştır. Tasarıma başlarken örnek bir fotoğraf çizim ortamına aktarılarak üzerine birebir çizim gerçekleştirilmiştir. Gerçekleştirilen çizime 20mm kalınlık verilerek üç boyutlu hale getirilmiştir. Uygulanan işlemler Şekil 5.14’de gösterilmiştir.

Kaba şekilde elde edilen antenin gövdesine 30mm çapında bir daire çizilerek 4mm radyus uygulanmıştır. Antenin balkon kısmına 12mm çapında bir silindir çizilerek 2mm et kalınlığı verilerek 10cm yükseklik kazandırılmıştır. Bahsedilen işlemler Şekil 5.15’de gösterilmiştir.



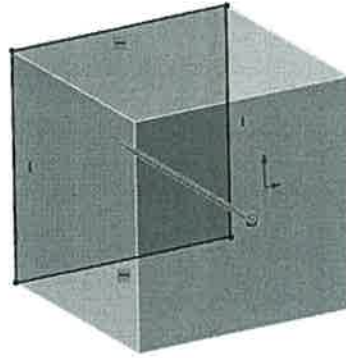
Şekil 5.15. Antene üç boyut kazandırma işlemi



Şekil 5.16. Anten çizimi

Antenin iç kısmı için birbiri içine geçecek şekilde 5 adet silindir tasarlandı ve bu silindirler birbiri ile montaj ilişkisi kullanılarak ilişkilendirildi. Anten tasarımının tamamlanmış hali Şekil 5.16'da gösterilmiştir.

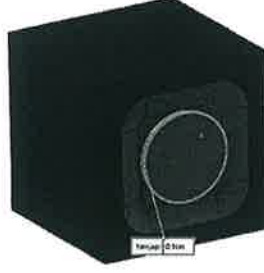
Uzayan boyun yapısı üzerinde kamera bulunan örümcek robotun taşıyacağı kameranın tasarımına başlarken ilk olarak 22x22mm kenar uzunluklarına sahip bir katı model oluşturulmuştur. Oluşturulan katı modele 22mm derinlik verilmiştir. Bu kısma kadar yapılan işlemler Şekil 5.17'de gösterilmiştir.



Şekil 5.17. Kamera taslak çizim

Kameranın ön kısmına çizim açılarak mercek çizimine geçilir. Ön plandan 1mm derinliğinde 18mmlik bir kare kesilerek alınmıştır. Oluşan boşluğun kenarlarına 5mm radius uygulanmıştır. Boşluğun merkezine çapı 8mm olan bir çember çizilerek çembere

1.8mm yükseklik kazandırılmıştır. İşlemler sonucunda elde edilen görünüm Şekil 5.18’de gösterilmiştir.

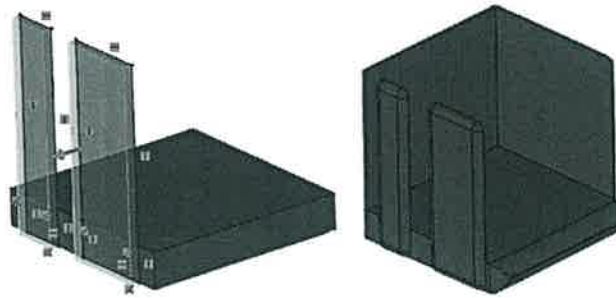


Şekil 5.18. Kamera mercek tasarımı

Merceğin kaba olarak çizimleri 3 boyutlu hale getirildikten sonra düğmeler, flaş ışıkları ve merceğin tam anlamıyla çizimi işlemleri yapılmıştır. Merceğin orta noktası referans alınarak 6 adet flaş ışığı 60 derece aralıklar ile dış tabaka üzerine çapı 1.2mm olacak şekilde çizilmiştir. Merceğin kendisi 5mm yükseklik verilerek katılaştırılmış ve 5mm radyus vererek ovalleştirilmiştir. Yapılan işlemler sonucu elde edilen görüntü Şekil 5.19’da gösterilmiş ve kameranın çizimi tamamlanmıştır.



Şekil 5.19. Kamera tasarımı



Şekil 5.20. Kamera tutacağı tasarım

Kamera ile anten son parçasının bağlantısını yapması için kamera tutacağı çizimine ihtiyaç duyulmuştur. İlk olarak 23mm kenar uzunluklarına sahip bir kare oluşturularak bu kareye 4mm yükseklik verilmiştir. Tutacağıın sol tarafına 20mm yüksekliğinde 2 adet

dikdörtgen çizilmiş ve 1mm et kalınlığı verilerek üç boyutlu hale getirilmiştir. Bu dikdörtgenlerin tek bir yapı olmamasının sebebi kameranın üzerindeki düğmelerin etkin kullanılabilmesini sağlamaktır. Arka ve sağ tarafa 20mm yüksekliğinde iki kenar çizilmiştir. Bu kenarlara 1mm et kalınlığı verilmiştir. Kamera tutacağıın ön tarafına düşmeyi önlemek için bir bölge tasarlanmıştır. Bahsedilen işlemler Şekil 5.20’de gösterilmiştir.



Şekil 5.21. Kamera montaj hali

Kameranın tasarımı ve kamera tutacağıın tasarımı tamamlandıktan sonra birbirlerine montaj edilmiş halleri Şekil 5.21’de gösterilmiştir.



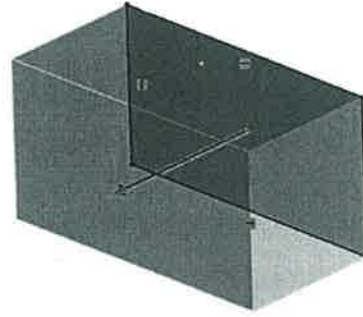
Şekil 5.22. Pabucun kesit görüntüsü



Şekil 5.23. Pabuç tasarımı



Robotun ayakularına monte edilecek ve yer ile temasını saėlayacak olan pabu tasarımı yapılmıřtır. 20mm yksekliėi ve 18mm geniřliėi olan bir dikdrtgen izilmiřtir. izilen dikdrtgene 5mm kalınlık verilerek  boyut kazandırılmıřtır.  boyut kazanmıř olan dikdrtgenin alt ve st kısımlarına 2.5mm radyus uygulanmıřtır. Kabası oluřan pabucun orta kısmına 6mm aplı bir silindir izilerek katılařtırılmıřtır. Btn bu iřlemler tamamlandıktan sonra ortaya izilen silindirin saė ve sol kenarları dıřa doėru 1.5mm uzatılmıř ve 3x1.5mmlik dikdrtgen haline getirilmiřtir. Tasarımı yapılan pabucun kesiti alınmıř hali Őekil 5.22’de gsterilmiřtir. Tasarımın tamamlanmıř hali Őekil 5.23’de verilmiřtir.



Őekil 5.24. Servo motor taslak izimi

Robot zerinde kullanılacak servo motorların tasarımı Hitec-Hs311 servo motoru zerine yapılmıřtır. Motorun boyut bilgileri alınarak gereėe uygun boyutlarda izimler yapılmıřtır. 40x21mm llerinde bir dikdrtgen izilerek ekstrzyon iřlemi ile 20mm ykseklik kazandırılmıřtır. Yapılan iřlem Őekil 5.24’de gsterilmiřtir.



Őekil 5.25. Servo motor mil izimi

Kaba izimi yapılmıř olan servo motora ykseklik kazandırıldıktan sonra 4 křesine 2mm radyus iřlemi uygulanır. Radyus iřlemi uygulandıktan sonra saė kapakıėın zerinden 10mm l alınarak 3mm apında daire zilir. izilen bu daireye 1.5mm ykseklik kazandırılır. Bylece servo motorun mili izilmiř olur. Bahsedilen iřlemlerin uygulanmıř hali Őekil 5.25’de gsterilmiřtir.



Şekil 5.26. Servo motor

Mil çizimi yapılmış olan servo motorun montaj sırasında şekil üzerine oturtulabilmesi için 2 tane pin çizilmiştir. Bunlar gerçek uygulamada bağlantı vidalarıdır. Pinlerin çizilmiş hali ve servo motorun çiziminin tamamlanmış hali Şekil 5.26'da gösterilmiştir.



Şekil 5.27. Servo-1 bacak aparatı taslak çizimi

Tasarımı yapılan robotun bir bacağına iki adet servo motor bulunacak şekilde tasarlanmıştır. Bacağın şasiye bağlandığı servo motorun bacak ucuna bağlantısını yapan bir aparat tasarlanmıştır ve bu aparata servo-1 bacak aparatı ismi verilmiştir. Aparatın bacak ile temas edeceği bölgeye 8mm çapında bir silindir çizilmiştir. Bu silindirin merkezinden yukarıya 35mm yükseklikte bir çizgi çizilmiştir. Takibin de sağa doğru 25mm uzunluğunda bir çizgi daha çizilmiştir. Çizginin bitiş noktasına çapı 5mm olan bir daire çizilmiş ve iki daire birbiri ile 3 nokta komutu ile birleştirilmiştir. Oluşturulan taslak çizime 2mm et kalınlığı verilmiştir. Yapılan işlemler Şekil 5.27'de gösterilmiştir.



Şekil 5.28. Servo-1 bacak aparatı

Taslak çizimi yapılan servo-1 bacak aparatının şasi bağlantısı olan servo ile olan kenarına 1.5mm çapında iki adet vida yuvası açılmıştır. Diğer kenara ise ikinci servo motorun milinin bağlanacağı 2mm çapında bir yuva açılmıştır. Tasarımı tamamlanan servo-1 bacak aparatının görüntüsü Şekil 5.28’de verilmiştir.

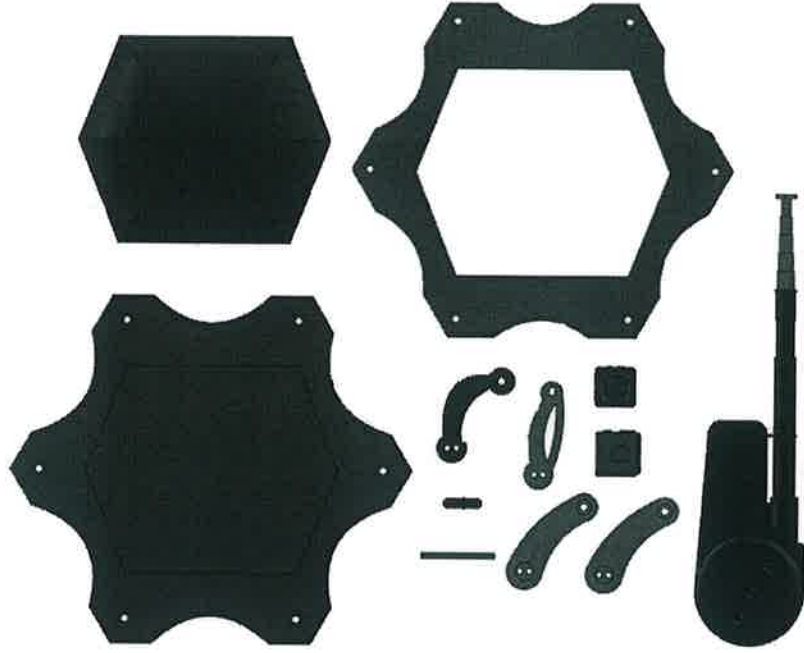


Şekil 5.29. Aparat-2 taslak çizim



Şekil 5.30. Servo-2 bacak aparatı

Robotun bir bacağına bulunan ikinci servo motorun aparatının çizimi için aralarında 45mm uzaklıkta birisi 8mm diğeri 6mm çapında iki adet çember çizilmiştir. Çizilen çemberler birbiri ile birleştirilerek taslak çizim elde edilmiştir. Çember çizimi ve taslak çizim Şekil 5.29’da gösterilmiştir. Taslak çizimi yapılmış olan aparatın çizimine ekstrüzyon ile 1.5mm kalınlık verilmiştir. Daha sonra üst ve alt daire içerisine 1.5mm çapında daireler çizilerek içerisi oyulmuştur. Böylece aparatın çizimi tamamlanmıştır. Tasarımı tamamlanan aparat-2 Şekil 5.30’da gösterilmiştir.



Şekil 5.31. Örümcek robot için tasarlanan parçalar

Tasarımı yapılan robotun tasarlanan tüm parçaları Şekil 5.31’de gösterilmiştir. Bu parçalar SolidWorks üç boyutlu tasarım programının montaj sekmesine girilerek birbirleri ile ilişkilendirilerek toplanır. Bu ilişkilendirme parçaların birbirleri olan temas noktalarının geometrisine göre farklılık gösterir. Matlab tabanında yapılacak olan simülasyon kontrolünü de doğrudan etkileyecek olan bu montaj ilişkileridir. Örneğin montajda eş merkezli komutu kullanılarak yapılan tüm ilişkilendirmeler simulink ortamında dönel eklem olarak görülür.



Şekil 5.32. Örümcek robot tasarımı

Şekil 5.32’de örümcek robot için tasarlanan parçaların montaj yapılmış hali görülmektedir. Tasarım aşamasından sonra bu tasarım kullanılarak kontrol aşamasına geçilmiştir.

## 5.2. Simulink Kontrol Aşaması

Tasarımını yaptığımız altı bacaklı örümcek robotun kontrol aşamasında Matlab'i kullandık. Matlab farklı programlama yöntemleri olan bir programlama dilidir. Bu çalışmada tasarlanmış bir şeklin kontrolünü yapacağımız için Matlabsimulink yöntemi ile kontrol yapılmıştır.

### 5.2.1. XML dosyası oluşturma

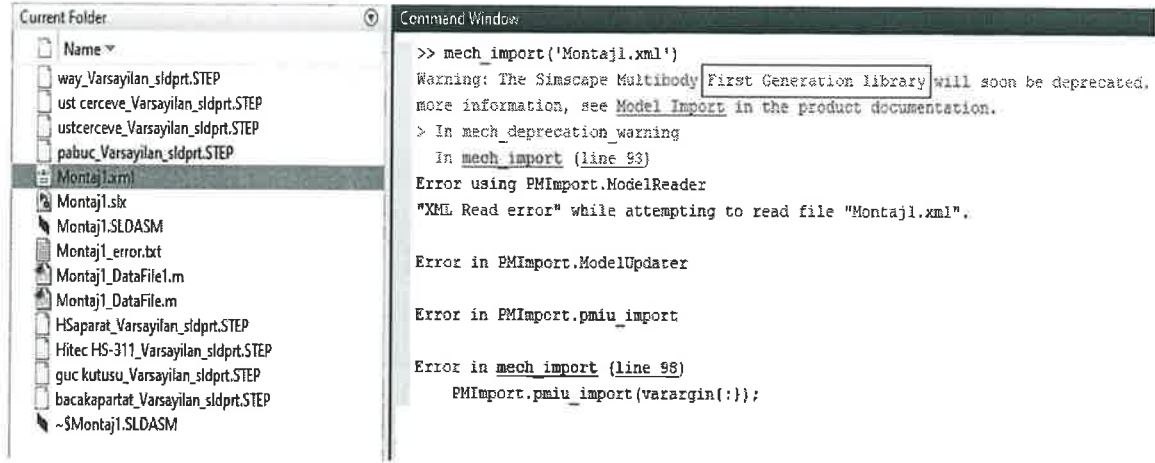
Kontrol aşamasına başlangıçta ilk olarak tasarım Solidworks ortamından Matlabsimulink ortamına aktarılır. Bu işlem için Solidworks ortamında yapılmış tasarımı, önceden eklentisini yapmış olduğumuz SimscapeMultibody uzantısı kullanılarak export yapılır. Export işlemi sonrasında oluşan sldprt uzantılı parça bilgisini içeren dosyalar ve montajın tamamına ait olan xml dosyası seçilerek bir klasör içerisine taşınır. Şekil 5.35'de export işleminden sonra oluşan veri dosyaları gösterilmiştir.

Ad	Tür	Boyut
bacakapartat_Varsayilan_sldprt.STEP	STEP Dosyası	60 KB
guc kutusu_Varsayilan_sldprt.STEP	STEP Dosyası	67 KB
Hitec HS-311_Varsayilan_sldprt.STEP	STEP Dosyası	114 KB
HSaparat_Varsayilan_sldprt.STEP	STEP Dosyası	42 KB
Montaj1.SLDASM	SOLIDWORKS Ass...	14.484 KB
Montaj1.xml	XML Belgesi	46 KB
Montaj1.slx	SLX Dosyası	86 KB
Montaj1_DataFile.m	M Dosyası	52 KB
Montaj1_error.txt	Metin Belgesi	1 KB
pabuc_Varsayilan_sldprt.STEP	STEP Dosyası	94 KB
ust cerceve_Varsayilan_sldprt.STEP	STEP Dosyası	204 KB
ustcerceve_Varsayilan_sldprt.STEP	STEP Dosyası	103 KB
way_Varsayilan_sldprt.STEP	STEP Dosyası	36 KB

Şekil 5.35. XML dosyası oluşturma

Şekil 5.35'de kırmızı çerçeve içerisinde gösterilmiş olan xmluzantılı montaj dosyasının, XML uzantılı oluşturmamızdaki amaç Matlab ortamında çağırabilmemizdir. Matlabsimulink ortamında oluşturulması için montaj dosyalarının xml uzantılı olması gerekir. Dosyalar oluşturulduktan sonra Matlab komut sayfasında xml uzantılı olan dosya çağırılır. Bu çağırma işlemi sırasında "mach\_import" komutu kullanılır. Fakat Matlab programlama dili her geçen yıl yenilendiği için çağırma sırasında bazı problemler yaşanabilir. Bu problemler kuşak farkından kaynaklanmaktadır. Yeni çıkan

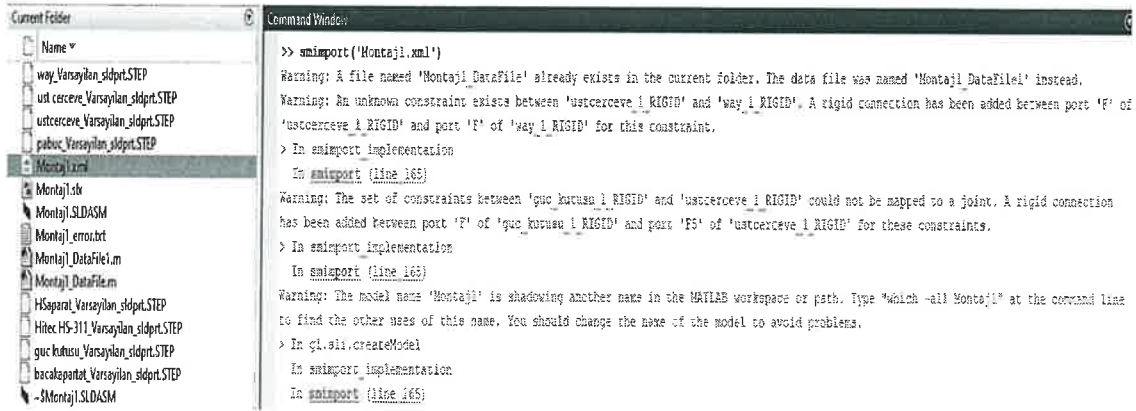
sürümlerkullanıldığında eski komutların çoğu kuşak hatası verebilmektedir. Montaj çağırma sırasındaki kuşak hatası Şekil 5.36’da gösterilmiştir.



Şekil 5.36. Jenerasyon hatası

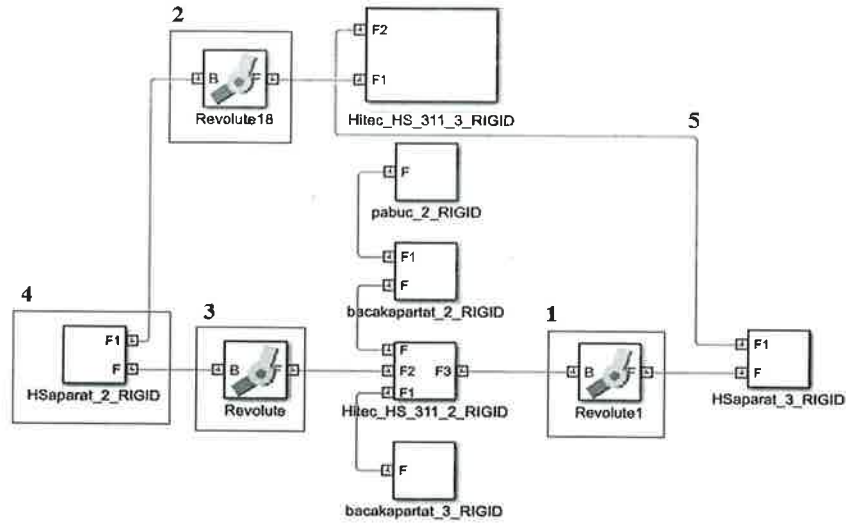
## 5.2.2. Montaj çağırma işlemi

Yeni nesil simulink de çağırma işlemi yapmak için “sm\_import” komutu kullanılır. Komuttan sonra çağırılacak olan xml dosyası uzantısı ile birlikte parantez içerisinde olacak şekilde komutun yanına eklenerek montaj çağırılır. İşlem Şekil 5.37’de gösterilmiştir.



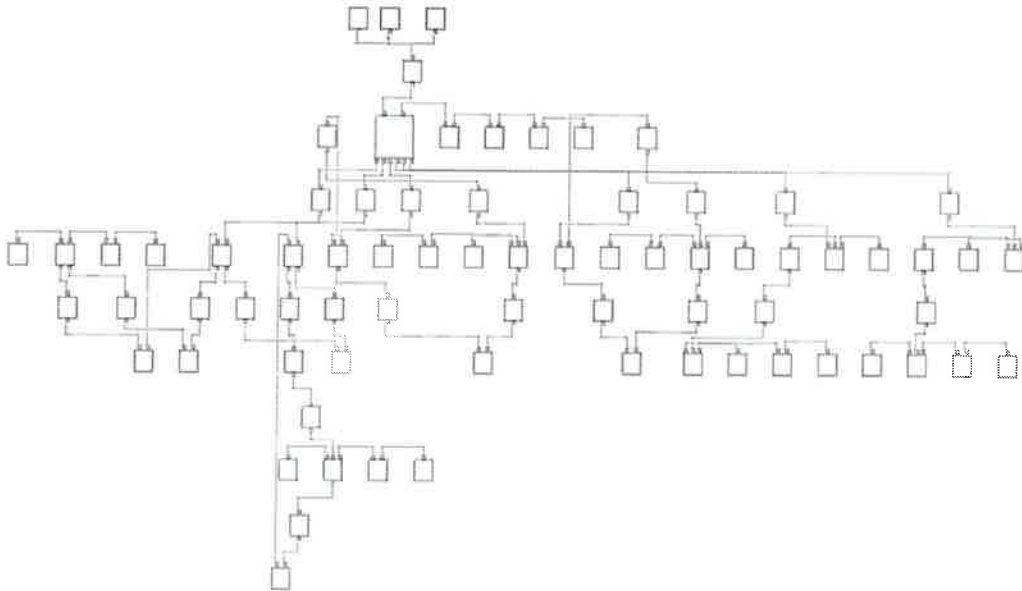
Şekil 5.37. Montaj çağırma işlemi

Çağırma işlemi başladığında montajın içerisindeki her parça ve parçaların birbirleri arasındaki olan montaj ilişkileri için arka planda tüm veriler otomatik olarak oluşur ve işlem tamamlandığında simulink kontrol paneli ekrana gelir. Ekrana gelen bu sayfada kullanılan her bir parçanın katı model olarak görünümünü içeren solid dosyaları, parçalar arasındaki hareketli bölgelerin her birinin kontrolünü sağlamaya yarayan eklemler(silindirik, prizmatik) ve mekanik ayarparametreleri bulunur.



Şekil 5.38. Tasarımın temel Matlab kontrol paneli görünümü

Çağırma işlemi sonrasında ekrana gelen görüntü Şekil 5.38’de gösterilmiştir. Şekilde kırmızı kutu içerisinde gösterilen 1 ve 3 numaralı yapılar eklem dönüşünü sağlayan Revolute birimleridir. 1 numaralı Revolute bacağın ilk eklemine kontrol eder. Yani HSapart yapısını kontrol eder. 2 numaralı Revolute ise servo motora bağlı olan eklem yapısıdır. Bu Revolute pabucun bağlı olduğu kısmı yani uç işleyiciyi kontrol eder. Bu yapıları ilerleyen bölümlerde Fuzzy ve PID kontrolörler ile kontrol edeceğiz. 4 numaralı kutuda gösterilen yapı katı model bilgisi içeren solid yapılarından bir tanesini göstermektedir. Solidin şekil üzerindeki isminden de anlaşıldığı üzere bacak da bulunan ikinci aparat bilgilerini içermektedir.



Şekil 5.39. Matlab tarafından oluşturulan düzenlenmemiş ilk temel devre

Şekil 5.38’de gösterilen devre sadece bir bacak için Matlab tarafından oluşturulan devredir. Tasarım hexapod olduğu için bu devreden altı tane oluşturulur. Matlab tarafından oluşturulan geneldevre üzerinde düzenleme yapılmadığı için karmaşa vardır ve bu durumda kontrol yapmak ve çalışmak oldukça güçtür. Temel devre Şekil 5.39’da gösterilmiştir.

Bu karışıklığı önlemek için her bacağın parametreleri Alt+G kısa yolu kullanılarak kendi içerisinde bloklaştırılır. Bloklaştırma sonrasında karmaşa ortadan kalktığı gibi giriş çıkışlarda blok dışarısına verilen isimde aktarılır. İlgili kısma bağlantı yapılacağı zaman bu pinlerden bağlantı yapılır.

### 5.2.3. Çözücü(solver) seçimi

Dinamik bir sistemi simüle etmek için, durumlarını belirtilen bir zaman aralığında ardışık zaman adımlarında hesaplamak gerekir. Bu hesaplama, bir sistem modeli tarafından sağlanan bilgileri kullanır. Zaman adımları, hesaplama gerçekleştiğindeki zaman aralıklarıdır. Bu zaman aralığının büyüklüğüne adım büyüklüğü denir. Bir modelin durumlarını bu şekilde hesaplama sürecini modelin çözülmesi olarak bilinir. Tüm sistemlere tek bir model çözme yöntemi uygulanmaz. Simulink çözücü olarak adlandırılan bir dizi program sunar. Her çözücü, bir modelin çözümünde özel bir yaklaşım içermektedir. Bir çözücü, modeli temsil eden adi diferansiyel denklem kümesini çözmek için sayısal bir yöntem uygular. Bu hesaplama sayesinde, bir sonraki benzetim adımının zamanını belirler. Bu başlangıç değeri problemini çözme sürecinde, çözücü de belirttiğiniz doğruluk gereksinimlerini karşılar. Simulink, iki ana çözücü tipi sunar: sabitlenmiş ve değişken adımlı çözücüler. Model yapılandırma parametrelerinin Çözücü bölmesinde her tipin altındaki çözücülerini görülebilir(Anonim).

Matlabsimulink ortamında kullandığımız her bir komut ne kadar görsel olarak görünse de arka planda matematiksel bir işlem olarak işlenmektedir. Birçok matematiksel çözücü seçeneği sunan Matlab de tüm çözücülerini kullanmak mümkün değildir. Bunun için yaptığımız çalışma için en uygun olan çözücü araştırıp kullanmak gerekir. Simulink çalışmalarında kullanılan en yaygın çözücü ode-45(Dormand-Prince)’dir.

Sayısal analizde, Dormand-Prince (RKDP) yöntemi veya DOPRI yöntemi, adi diferansiyel denklemlerin çözümü için açık bir yöntemdir. Yöntem, ODE çözenlerin Runge-Kutta ailesinin bir üyesidir. Daha özellikli olarak, dördüncü ve beşinci sıradaki doğru çözümleri hesaplamak için altı fonksiyon değerlendirmesi kullanır. Bu çözümler arasındaki fark daha sonra (dördüncü dereceden) çözümün hatası olarak kabul edilir. Bu hata tahmini, uyarlamalı kademeli uyum algoritmaları için çok uygundur(Dormand ve ark., 1980).



Dormand-Prince yönteminin yedi aşaması vardır, ancak FSAL (En Son Aynı İlk) özelliğine sahip olduğu için her adımda yalnızca altı fonksiyon değerlendirmesi kullanır: son aşama bir sonraki adımın ilk aşaması ile aynı noktada değerlendirilir. Dormand ve Prince, beşinci sıradaki çözümün hatasını en aza indirmek için yöntemlerinin katsayılarını seçtiler. Dördüncü dereceden çözümün küçük bir hata yapması için inşa edilen Fehlberg metodu ile temel fark budur. Bu nedenle, yerel ekstrapolasyon olarak bilinen bir uygulama olan entegrasyona devam etmek için yüksek dereceli çözüm kullanıldığında Dormand-Prince yöntemi daha uygundur (Shampine, 1986.Nørsett ve ark.,2008)

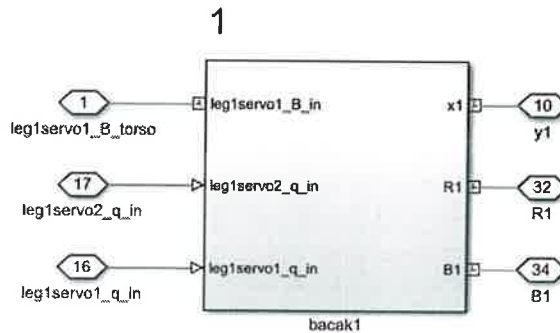
Dormand-Prince şu anda Matlab'in ode45 çözücüsünde varsayılan yöntemdir ve Simulink'in model gezgini çözücüsü için varsayılan seçimdir. DOPRI5 adlı algoritmanın Fortransız yazılım uygulaması da mevcuttur(Anonim).



Şekil 5.40. Simulink çözücü seçimi

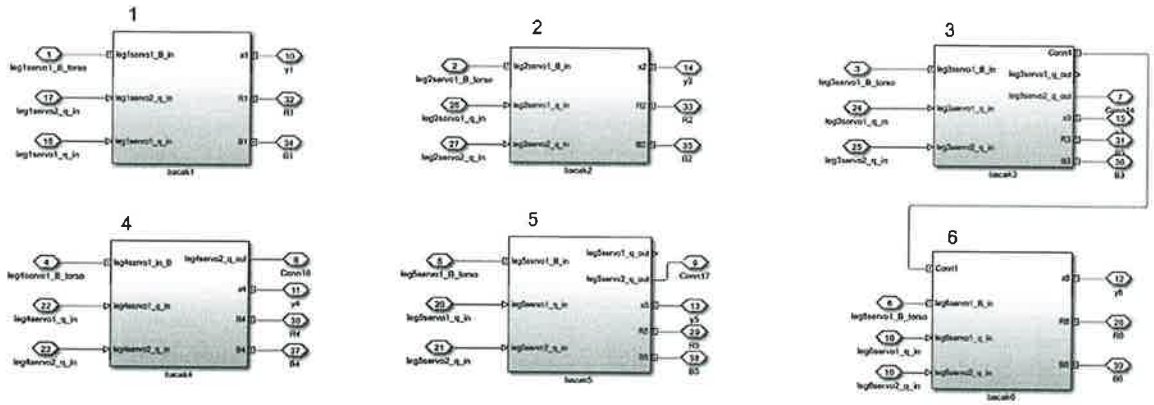
Simulink için kullanılan ode-45 Dormand-Princemethodunun çözücü olarak seçimi Şekil 5.40'da gösterilmiştir. Seçimi yapabilmek için çözücü tipinin Variable-Step olması gerekir.

#### 5.2.4. Blok oluşturma



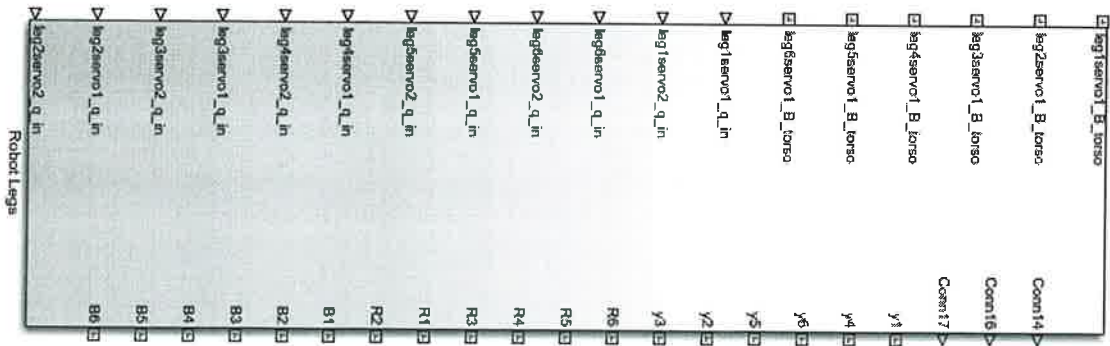
Şekil 5.41. Hexapod robotun 1 numaralı bacak bloğu

Şekil 5.41’de gösterilen blok hexapod robotun 1 numaralı bacağının parametrelerinin bulunduğu bloktur. Bu blok kısa yolun kullanılarak düzenlenmenin yapılmasından sonra elde edilir. Elde edilen blok ilgili bacağın adı ile adlandırılır. Daha sonra blok üzerinde görülen giriş çıkışlara çıkışın ismi verilir. Çalışma kolaylığı açısından bunlar gerekli adımlardır.



Şekil 5.42. Hexapod bacak blokları

Şekil 5.42’de verilen bloklar sırası ile kontrolünü yapacağımız hexapod robotun bacaklarını temsil eden bloklardır. İçerilerinde bulduklarını parametreler eklemler ve solid yapıları ile simulink oluşumunu sağlamaktadırlar. Bu blokları da bir blok içerisinde toplamak gerekir. Tüm bloklar kendi içerisinde tekrar toplanmak zorundadır.

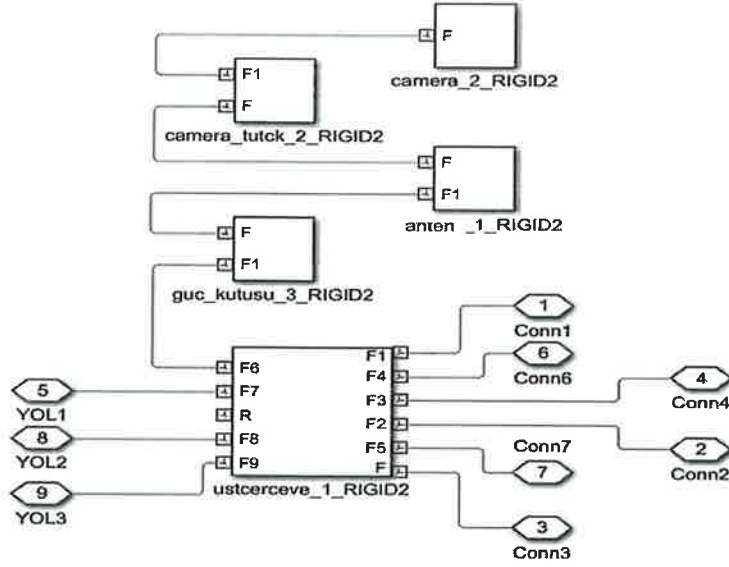


Şekil 5.43. Hexapod robotun bacak bloğu

Tüm bacakların toplandığı blok Şekil 5.43’de gösterilmiştir. Bacaklarla ilgili işlemler için bu blok içerisine girerek ilgili bacağa direk müdahale edilebilmektedir.

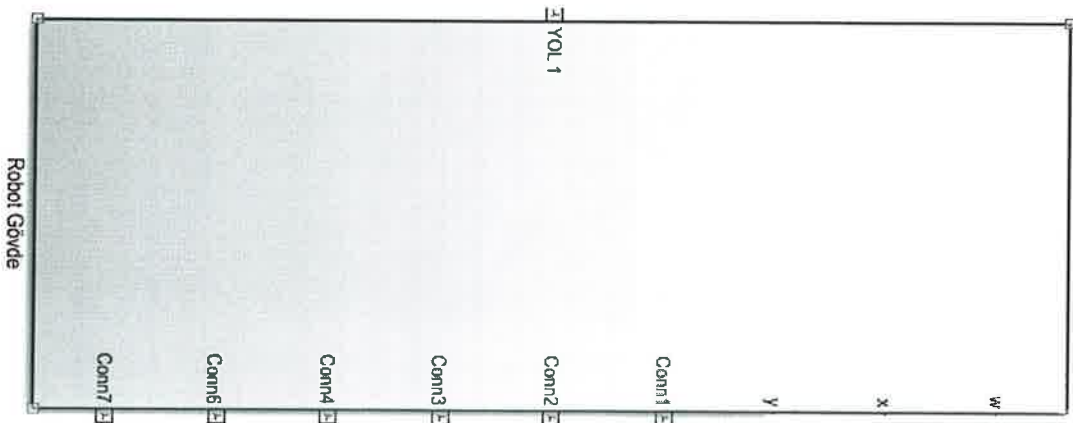
Robotun diğer parçalarını içeren bloklarda oluşturmak gerekir. Bunlar kamera, şase kamera tutacağı ve kamera anteni gibi parçalardır. Bu kısımlarda herhangi bir kontrol

yapmayacağımız için tüm bileşenleri bir tek blok içerisinde toplamak mümkündür. Belirtilen parametreler Şekil 5.44'de ve parametrelerin tek bir blok içerisinde toplanmış hali Şekil 5.45'de gösterilmiştir. Şekilde görülen parçalar arasındaki yeşil yollar parçalar arasındaki bağlantıları temsil eder.



Şekil 5.44. Hexapod robot gövde yapıları bloğu

Şekilde üst çerçeveye giriş olarak görülen yol1, yol2 ve yol3 parametreleri ilerleyen bölümlerde sürtünme tanımlama konusunda anlatılacaktır. Kamera tutacak solid yapısından kameraya çıkan yeşil yolu ele alacak olursak, burada kamera ve tutacak arasındaki bağlantı anlatılmıştır. İki nesne arasında tasarım ortamında bir montaj ilişkisi tanımlanmıştır. Fakat bu tanımlamada bir eklem yoktur. Yani bu iki nesne arasındaki bağlantı sabittir. Herhangi bir hareket yoktur. Bunu direkt olarak 2 nesneyi birbirine yapıştırmaya benzetebiliriz.



Şekil 5.45. Hexapod robotun gövde bloğu

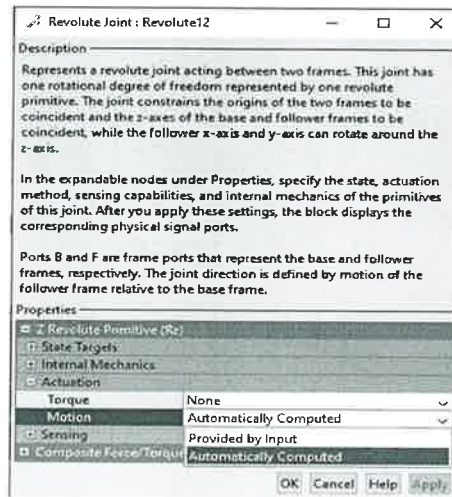
### 5.2.5. Simulinkte çağırılan montajın parametre ayarı

Çağırma sonucu gelen şemada düzenlemeler yaptıktan sonra “run” komutu ile çalıştırdığımızda Şekil 5.46’da görüldüğü gibi robot simulink ortamında karşımıza gelir. Fakat fark edildiği üzere tüm bacaklar düzensizdir ve sürekli rastgele yönlerde hareketler yaparlar. Bu düzensizliğin ve hareketin sebebi başlangıçta tüm eklemleri Matlab’in otomatik olarak düzensiz ve sınırsız bir şekilde kontrol etmesidir.



Şekil 5.46. Çağırma sonrası ilk görünüm

Şekil 5.38’de gösterilen Revolute yapılarında herhangi bir dış etki yoktur. Revolute blokları bağlantıların arasında etkisiz bir şekilde yer almaktadırlar. Dış kontrol gerektiğinde eklem içerisine girilerek parametrelerden ”motion seçeneği “provided by input” olarak ayarlanır. İşlem Şekil 5.47’de gösterilmiştir.



Şekil 5.47. Eklem hareket kontrol girişi ayarı

Şekil 5.47'deki işlem her eklem için uygulandıktan sonra Şekil 5.38'de görülen eklem yapılarının hepsi Şekil 5.48'de şekilde görüldüğü gibi elde edilir. Elde edilen bu yeni girişlere kontrolör çıkışları verilerek kontrol yapılır.

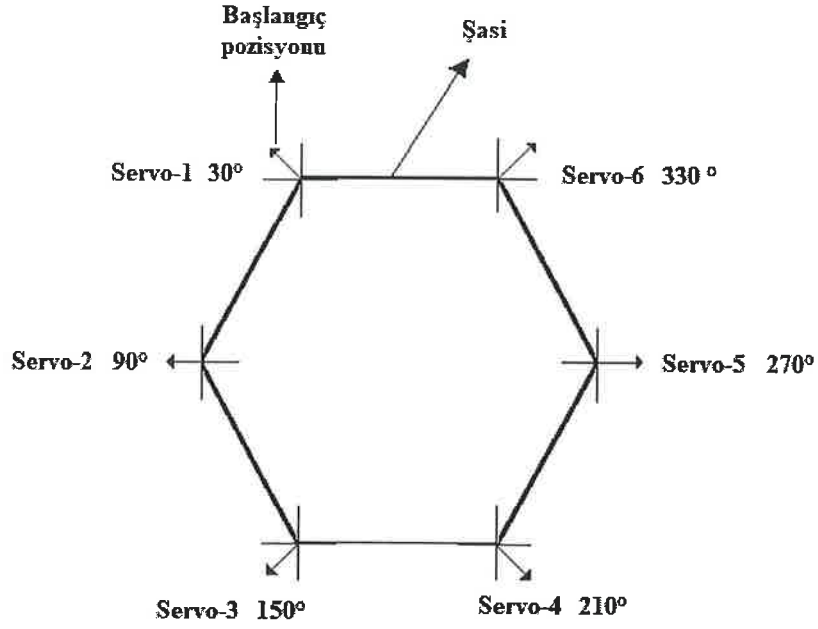


Şekil 5.48. Eklem kontrol pini

Eklem kontrollerini manüel hale getirdikten sonra Şekil 5.46'da bahsedilen rastgele hareketler engellenmiş olur ve şekil oluşturulduğunda bacaklar sınırlanmamış bir düzensizlikte hareketsiz olarak görülür.

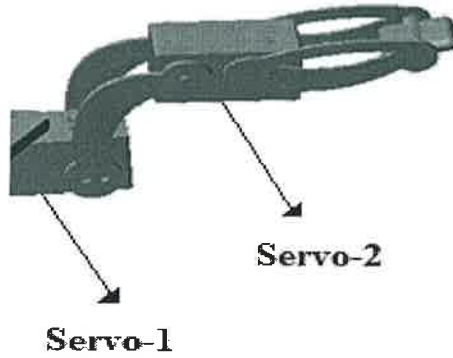
#### 5.2.6. Bacak eklemlerine pozisyon ve sınırlama işlemi yapımı

Eklemleri sınırlandırmak için her bir bacağına sabit bir değer aralığı ve sabit bir başlangıç pozisyon değeri girişi yapılır. Bu değerleri girmekteki amaç girilen bacağın başlangıç pozisyon derecesini belirlemek ve bacağın hareket açısını kısıtlamaktır. Bu kısıtlama kullanılacak olan servo motorun hareket açısına bağlı olarak değişebilir. Bu tezde 60 derecelik servo motor kullanımına göre işlem yapılmıştır. Bu işlemin her bacak için yapılması gerekir.



Şekil 5.49. Bacaklar arası açı aralıkları

Şekil 5.49'da şasi üzerine yerleştirilen servo motorların başlangıç konum dereceleri gösterilmiştir. Başlangıç konum değerleri robotun enerjilendiği anda ayaklarını çekmek isteyeceği konumu belirtmektedir. Gösterilmiş olan dereceler bacaklarda bulunan birinci servo motorlar için geçerlidir. Açı saat yönünün tersine taranmaktadır. Her bacak arasında 60 derece vardır. Hexapod bir tasarım olduğu için 360 dereceyi tamamlamaktadır. Her bacakta iki eklem bulunması her bacakta iki servo motor olduğu anlamına geldiği için her bacak için iki farklı başlangıç pozisyonu belirlenmesi gerekir. Bu tezde yapılan kontrol de her bacağın ilk motoru şasiye bağlı olandır ve hareketi sağlayan bu servo motorlardır. Birinci servo motorların üç konumu vardır bunlar başlangıç pozisyonu, ileri hareket konumu ve geri hareket konumudur.



Şekil 5.50. Tasarım üzerinde bir bacağın servo motorları

İkinci servo motor ise dizde bulunmaktadır. Oturma kalkma işlemini yapmaktadır. İkinci servo motorların üç konumu vardır. Bunlar başlangıç pozisyonu, bu pozisyon ayakta kalma pozisyonudur. İkinci konum ise yürüme pozisyonudur, bu pozisyon ise yürüme esnasında ilgili ayağı havaya kaldırarak sürtmeyi engeller. Üçüncü pozisyon ise oturma pozisyonudur. Robot yeri tutmak için bu pozisyonu kullanır. İşlem tamamlandıktan sonra robotun bacak pozisyonları Şekil5.51’de ve tek bir bacak yapısı Şekil 5.50’de gösterilmiştir.

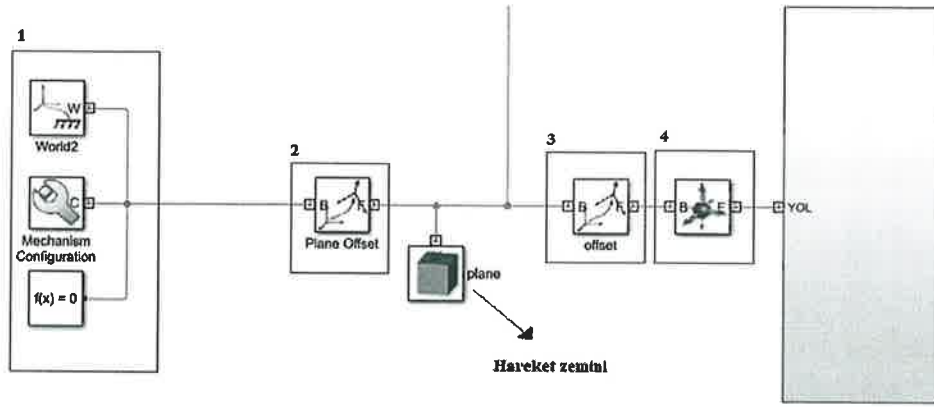


Şekil 5.51. Robotun bacak başlangıç pozisyonları

### 5.2.7. Mekanik parametreler ve uzay ortamında konum belirleme

Simülasyon ortamında mekanik parametrelerin kontrol edildiği komutlar şekil parçalarına doğrudan bağlanır. Bu parametreler şekle yer çekimi, x-y-z ekseninde konum belirleme ve 6 dereceli serbestlik derecesi eklemegibi etkilerde bulunur.

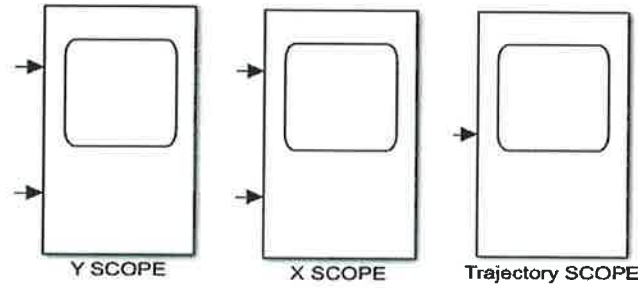
Şekil 5.52’de mekanik parametreler gösterilmiştir. 1 numaralı gösterilen birimler robotun ana parametreleridir. Çözücüler, yer çekimi gibi ayarlar bu birimler üzerinden yapılır. 2 numaralı kutuda gösterilen “PlaneOffset” isimli birim ise robotu yürüteceğimiz platformun uzay boşluğundaki konumu belirler. Hareket zemini Plane isimli bir solid komutu ile oluşturulmuştur. Bu platform Matlab ortamında oluşturulmuş olup herhangi bir 3D tasarım programı kullanılmamıştır. 3 numaralı kutuda gösterilen ikinci Offset ise robotun platform üzerindeki konumunu belirlediğimiz birimdir. 4 numaralı kutucuk ise robota uzay ortamında 6 serbestlik derecesi tanımlayan “6-DOF Joint” komutudur. Tüm bu parametreler doğrudan program tarafından ana parça olarak görülen şasiye bağlanır. Bu işlemlerin eğik düzlemde testler yapabilmek için oluşturulan her yol için yapılması gerekir. Bu konu ilerleyen bölümlerde ayrıntılı anlatılacaktır.



Şekil 5.52. Mekanik parametreler

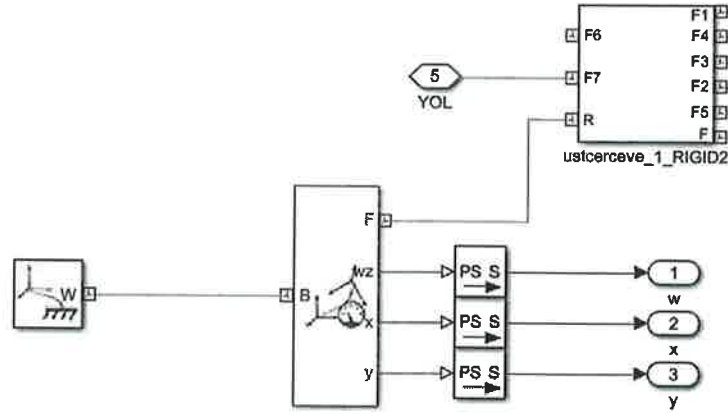
### 5.2.8. Robot konum kontrolü

Robotun platform üzerindeki hareketini x-y-z eksenlerinde kontrol etmek gerekir. Bu kontrolü sağlayabilmek için Şekil 5.54’de gösterilen “Transform Sensor” komutu kullanılır. Bu komutun çıkışında alınan veriler PS-S converterler ile simulink ortamına aktarılmaya uygun hale getirilir. TS üzerinden kontrol edilecek veriler seçilebilir. Tezde kullanılan algılayıcı ile x-y eksenleri ve açısal hız kontrol edilmiştir. Bu çıkışlar doğrudan bir her çıkışın kendi ilgili Scopebirimineaktarılmıştır.



Şekil 5.53. X-Y Trajectory Scopeları

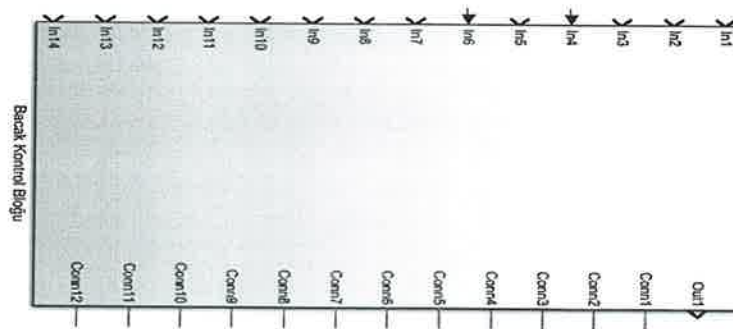




Şekil 5.54. Robot konum takip bloğu

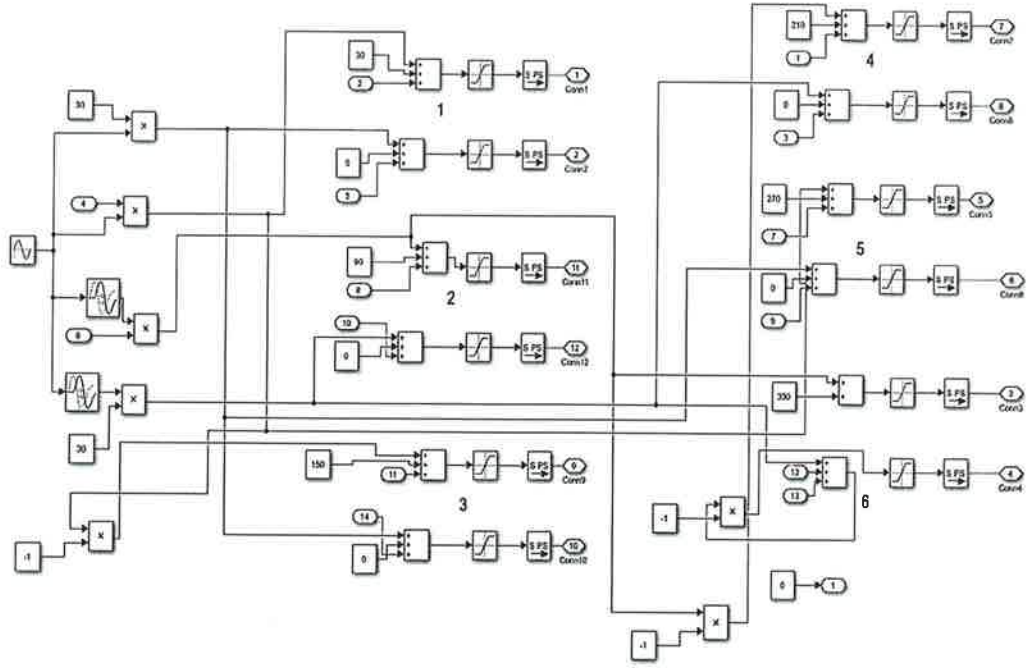
### 5.2.9. Bacak kontrol bloğu

Robotun bacakları, doğadaki yürüyüş şekillerinden, esinlenilerek hareket ettirilmiştir. En hızlı yürüyüş şekli olan tripod yürüyüşün uygulandığı bu tezde 1-3-5 numaralı bacaklar aynı dönem de geriye kalan 2-4-6 numaralı bacaklar ise diğer bir aynı dönem de hareket ederler. Yani ilk dönem de verilen ilk üç bacak hareket eder ikinci dönem de kalan üç bacak hareket eder ve hareket iki dönem de tamamlanmış olur. İki dönem arasında bir gecikme olması gerekir. Bu durumda Bacakların hareketleri ve bacakların kendi aralarındaki gecikmelerin sağlandığı bir bloğun oluşturulması gerekir. Simülasyon da bu amaç için kullanılmak üzere oluşturulan blok Şekil 5.55’de gösterilmiştir.



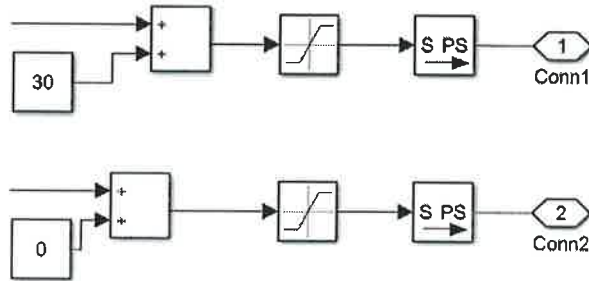
Şekil 5.55. Bacak kontrol bloğu

In6 ve In4 numaralı girişler kontrolör (PID veya Fuzzy) girişleridir. Conn-1 ve Conn-12 dahil bu aralıktaki çıkışlar ise doğrudan bacak bloğu içerisinde bulunan servo motorlara bağlanmaktadır.



Şekil 5.56. Bacak kontrol bloğu iç görünümü

Şekil 5.56'da gösterilmiş olan blok iç görünümünde her bacağın kendi kontrol birimi numaralandırılmış şekilde gösterilmiştir. 1 numaralı bacak örnek olarak Şekil 5.57'de gösterilmiştir.

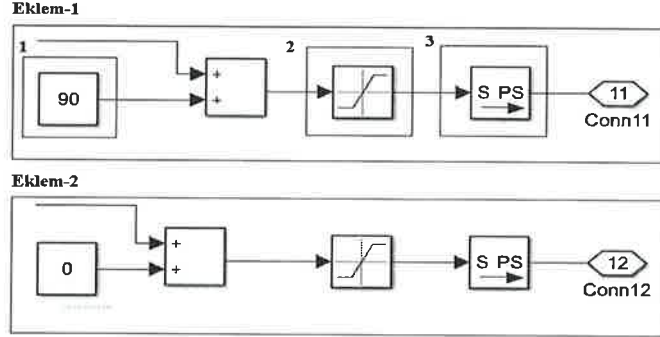


Şekil 5.57. Birinci bacağın kontrol bloğu

Şekil 5.55'de gösterilen blok içerisinde bacakların birlikte çalışmalarını sağlayacak olan parametreler bulunur. 1-3-5 numaralı bacakların ilk periyotta 2-4-6 numaralı bacakların ikinci periyotta hareket edeceğinden bahsetmiştik. Her bacakta iki eklem bulunduğu için her eklem için kontrolü ayrı yapılır. Şekil 5.58'de ikinci bacağın eklemleri gösterilmiştir. Bir numaralı kutu ile gösterilen değer ile robotun ilgili bacağının başlangıç pozisyonu belirlenir. 2 numaralı birim ise saturation komutudur. Bu komut ile robotun bacağının açılı aralığı(0-60) sınırlandırılır. 3 numara ile gösterilen birim ise

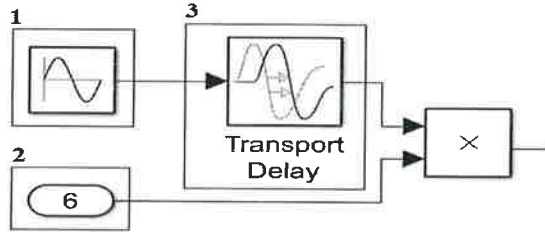
çeviricidir. Bu komutun görevi gelen verileri simulink ortamına çevirmek veya tam tersini yapmaktır.

Periyotlar arası gecikme oluşturmak için gecikme işlemi 2-4-6 numaralı bacakların birinci eklemlerine uygulanır. İkinci eklemlere uygulanan gecikmelerin bacak hareket periyotları arası süreler etkisi yoktur.



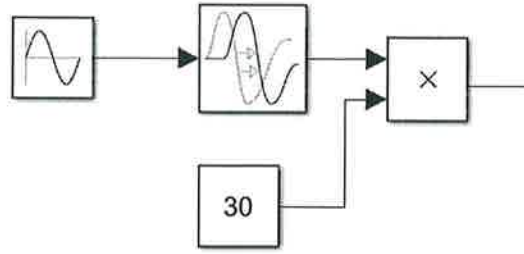
Şekil 5.58. İki numaralı bacağa ait kontrol bloğu

Hareket dönemleri arasında oluşan süre farkını meydana getirdiğimiz bölüm Şekil 5.59’da gösterilmiştir. 1 numaralı kutu ile gösterilen Sinüs sinyali 3 numaralı kutuda gösterilen “Transport Delay” komutu ile eklem 1 saniye gecikmeye uğratılır ve daha sonra bu gecikmeli sinyal ile In6’dan gelen kontrolör girişi çarpılarak elde edilen sinyal doğrudan ekleme uygulanır. Bu işlem sonucunda bu eklem kontrol ettiği bacak belirlenen gecikme süresi kadar gecikmeli çalışır. Bu işlem 2-4-6 numaralı bacaklar için yapılır. 1-3-5 numaralı bacaklara gecikmeli olmayan sinüs sinyali kontrolör girişi ile çarpılarak doğrudan uygulanır.



Şekil 5.59. Periyotlar arasına gecikme süresinin uygulanması

Şekil 5.58’de gösterilen ikinci eklem giriş sinyali Şekil 5.59’da gösterilmiştir. Görüldüğü gibi sinyal gecikmeye uğramıştır. Fakat bu sinyal gecikmesi bacağın kendi içerisindeki periyotları geciktirmek için verilmiştir. Birinci eklem hareket etmeyi sağladığı için ikinci eklem bu esnada ayağı havaya kaldırır ve hareket sonucunda bacağı yere indirir. Böylece yürüme hareketi sağlanmış olur. Bu zaman farkını oluşturmak için ikinci bacağa gecikme süresi uygulanır.

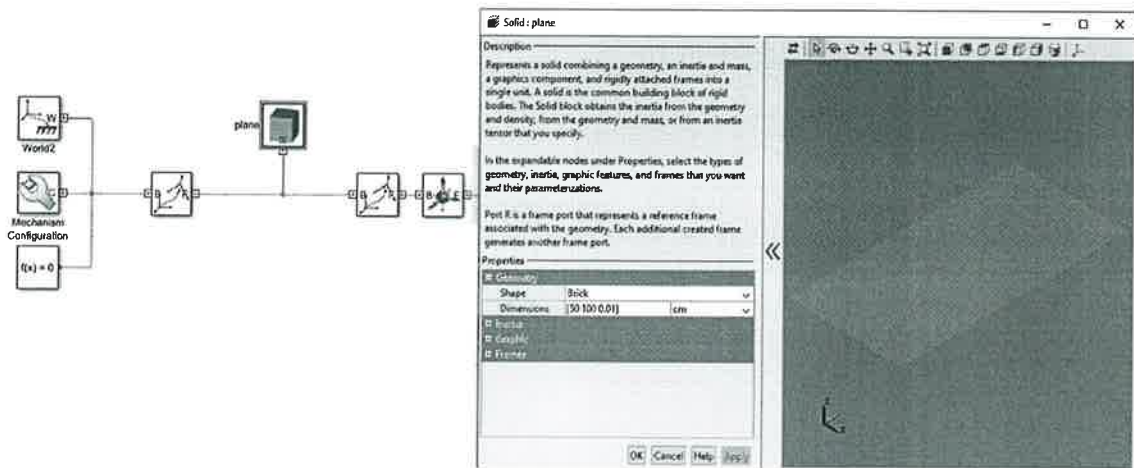


Şekil 5.60. İkinci eklemlerin giriş sinyali

Tüm bu işlemler yapıldıktan sonra ilgili sinyal Şekil 5.60'da gösterilen ilgili bacak girişine uygulanır. Bu girişler blok içerisinde bulunan eklemlere uygulanır. Birinci eklem sinyali ilk servo motora ikinci eklem sinyali ise ikinci servo motora uygulanır.

### 5.2.10. Uzak boşluğunda X-Y-Z eksenli düzlem tanımlama

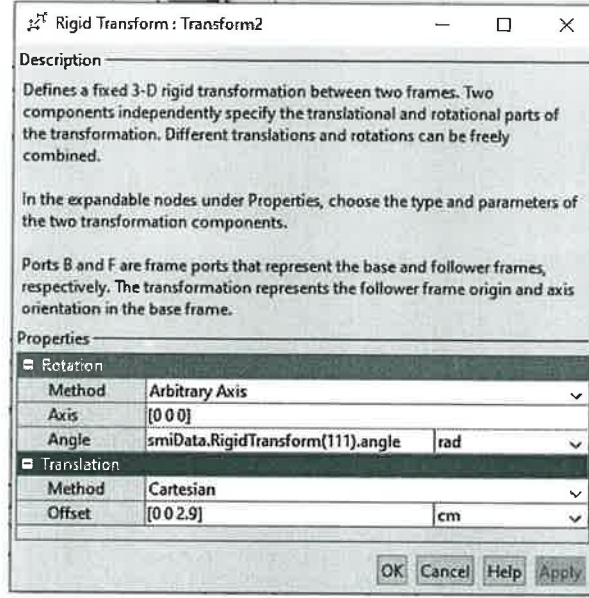
Robotun hareket blokları oluşturulduktan sonra hareket edebilmesi için bir düzleme, yola ihtiyaç vardır. Bu yol Matlab ortamında solid komutu kullanılarak oluşturulur. Fakat düzlemin oluşturulması robot ve düzlem arasında bir bağlantı tanımlanmadığı sürece Matlab tabanında anlamsızdır. Bu bağlantı robot ve yol arasındaki sürtünme ve yolun robota uygulayacağı tepki kuvvetidir. Solid komutu ile yol oluşturulması ve yolun ayar parametrelerini Şekil 5.61'de gösterilmiştir.



Şekil 5.61. Yol oluşumu ve parametre ayarları

Yol ve robotun uzak ortamındaki yerlerini transform komutları ile ayarlanır. Robot yolun üzerine gelecek şekilde konumlandırılır. Burada yolun ölçüleri cm cinsinden 50x100x0.01cm olarak ayarlanmıştır. Uzak boşluğunda konum ayarı yapılırken ilk olarak transform bloğu ile yolun konumu belirlenir. Yolun konumu belirlendikten sonra yol eklenir. Bu program mantığında ilk önce konumun eklendiği anlamını taşımaktadır.

Solid aracılığı ile yol eklendikten sonra ikinci transform bloğu eklenir. Bu robotun yol üzerindeki konumunu gösteren transformdudur. Bu transform bloğu robotun yol üzerindeki konumunun ayarını yapmak için kullanılır.

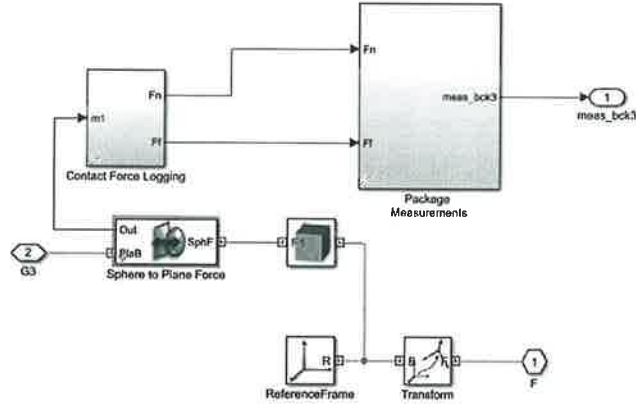


Şekil 5.62. Transform bloğu parametreleri

Şekil 5.62’de görülen “smiData.RigidTransform” terimi o parametrenin belirtilen adresten çağırıldığını gösterir. Bu parametreler bizim şekil oluştururken Matlab tarafından oluşturulan parametrelerdir. Offset alanında 2.9cmlik yükseklik ayarı görülmektedir. Bu değer robotun başlangıçta düşeceği yüksekliği göstermektedir.

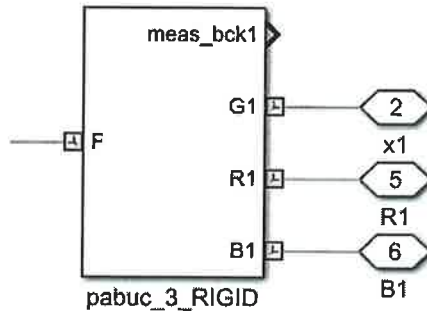
### 5.2.11. Sürtünme ve yüzey tepki kuvveti tanımlama

Yol oluşturmak ve robotu yolun tam üzerine konumlandırmak ikisi arasında bağlantı oluşturmak için yeterli değildir. Aslında Matlab ortamında yol oluşturmak bir diğer deyişle sadece gösteriştir. Robotun hareket alanında sürtünme ve zemin tepkisi oluşturmak için robotun ayaklarında bulunan her bir pabuç için farklı bir komut kullanmak gerekir. Bu komut Şekil 5.63’de görülen “Plane Force” komutudur.



Şekil 5.63. Robot ile yüzey arasında sürtünme oluşturma

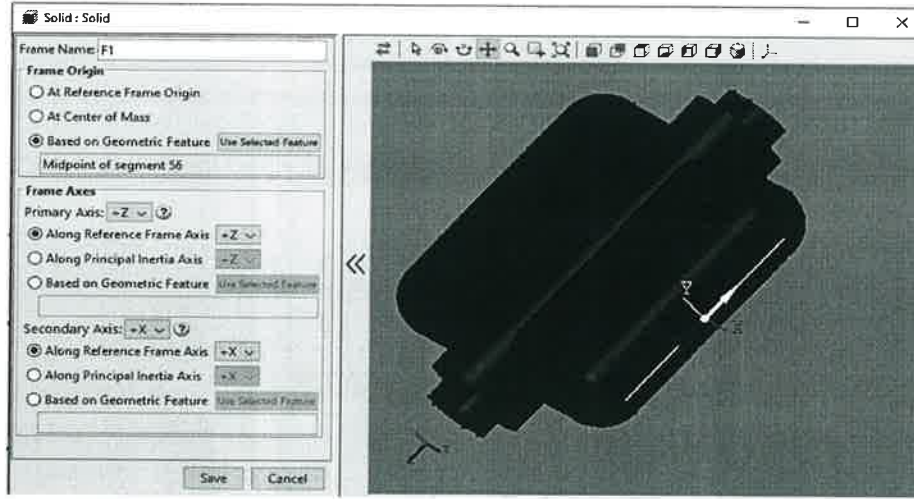
Bu parametre robotun ayaklarının zemine temas ettiği her noktaya yerleştirilmesi gereken bir parametredir. Komut temas noktasına bir çember çizerek bu çember ile yüzey arasında sürtünme kuvveti ve yüzey tepki kuvveti oluşturur. Oluşturduğumuz yolun sürtünme ile doğrudan bir ilişkisi yoktur. Komutun içerisine girildiğinde düzlem uzunluğu genişliği ve merkezi temas noktası olan çemberin çapının değerleri girilir.



Şekil 5.64. Her bacak için kullanılan pabuç bloğu

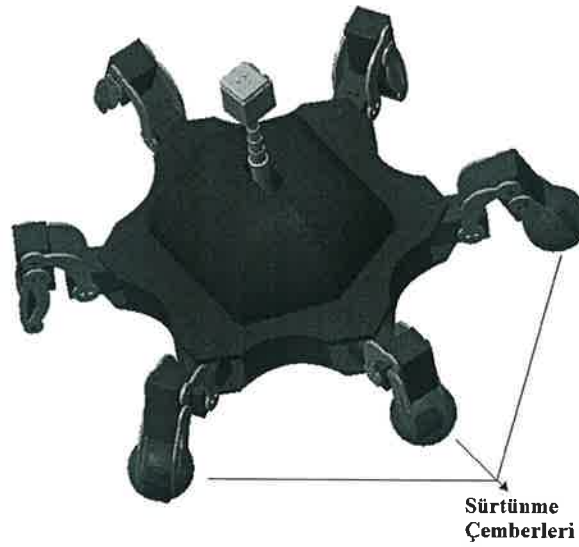
Burada girilen yol değerleri ile oluşturduğumuz yol arasında yine bir ilişki yoktur. Fakat iki yolun uzunluk ve genişlik değerleri aynı girildiğinde solid ile oluşturmuş olduğumuz yolun üzerine ince bir halı serilmişçesine görülen bir sürtünme yüzeyli görüntü elde edilmiş olur. Robot hexapod olduğu için toplamda 6 adet pabuç vardır. Bu işlem her pabuç için yapılmak zorundadır.

Pabuçların yüzey ile sürtünme noktasını ayarlamak için her ayağın kendi içerisinde bulunan pabuç solidi içerisine girilir. Buradan çerçeveler(Frames) kısmından seçili olan tik iptal edilir ve ana şekil ile olan bağlantı kaldırılır daha sonra yeni çerçeve eklenir. Eklenen yeni çerçevenin şekil üzerine doğrudan bağlantısı yapılır. Yeni çerçevenin özellikler kısmına gelerek yer ile olan temas noktası seçilir. İşlem Şekil 5.65'de gösterilmiştir.



Şekil 5.65. Pabuçlara sürtünme noktası belirlemek

Tüm pabuçlara sürtünme noktası ayarladıktan sonra Plane Force komutunun görünüm kısmından görseli aktif et tiki seçilirse yüzey ile robot pabuçları arasındaki sürtünme görülmüş olur. Şekil 5.66'da yüzeye temas edecek olan sürtünme çemberleri gösterilmiştir.



Şekil 5.66. Pabuçlara sürtünme çemberi eklemek

Şekil 5.61'de görülen görüntüde Plane önüne geri besleme bağlantısı yapılır. Bu geri besleme bağlantısı gövdeden gelir. Asıl olarak pabuçla bağlantılı olan bu geri besleme bağlantısı dolaylı yoldan gövde ile aktarılır. Bu bağlantının amacı yol ile robot arasındaki ilişkiyi kurmaktır.

Plane Force komutu ile tanımladığımız sürtünme ve yüzey tepki kuvveti eklediğimiz çemberlere etki eder. Sürtünme ve tepki kuvvetleri bizim girdiğimiz değerlere göre uygulanır. Yetersiz kuvvet değerleri girildiğinde robot yolda tutunamaz ve uzay boşluğuna düşer. Parametre değer girişleri ilerleyen bölümlerde anlatılacaktır. Bahsedilen olay Şekil 5.67’de gösterilmiştir.

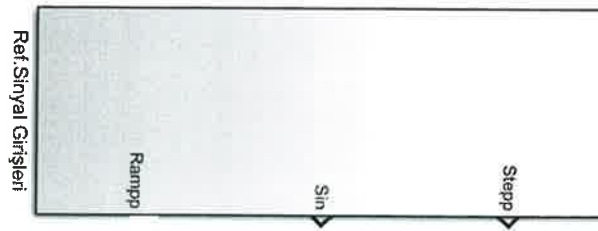


Şekil 5.67. Robotun yüzeye tutunamaması olayı

### 5.2.12. Sinyal giriş bloğu oluşturma

Tezin genel içeriği itibariyle robotun bir platform üzerinde istenilen noktalara istenilen kontrolör ve sürede hareketini sağlamak amaçlanmıştır. Bu hareketi sağlamak için bir giriş sinyali gerekir. Bu sinyal dalga kare veya düz çizgi sinyali olabilir. Robot her sinyali başarı ile takip edebilmelidir.

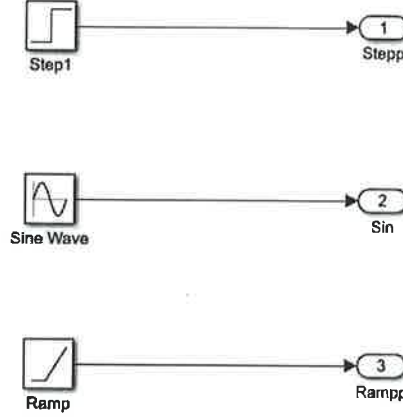
Robot X ekseninde sürekli ilerleme kaydeder yani yürüme işlemini gerçekleştirir ve ters yönde yürümüş olsa dahi bu negatif bir etki yapmaz. Fakat Y ekseninde ilerleme veya gerileme kaydedebilir. Y eksenini yöreğe belirleme ile kontrol edileceği için sinyal bloğu ile bir bağlantısı yoktur. Sinyal bloğu X eksenindeki ilerlemenin nasıl bir ilerleme olacağını belirlemek için oluşturulur.



Şekil 5.68. Sinyal giriş bloğu



Bu tez için oluşturulan sinyal bloğu içerisinde Sinüs sinyali(Sin) başlangıç noktası belirlene bilen Rampa sinyali(Rampp) ve değerlerin belirlenebildiği step sinyalini içeren bir giriş sinyal bloğu oluşturulmuştur. Hareket aşamasında etkin olarak rampa sinyali kullanılmıştır. Oluşturulan bloğun içeriği Şekil 5.69’da gösterilmiştir.



Şekil 5.69. Sinyal giriş bloğu içeriği

### 5.2.13. Yörünge(Trajectory)oluşturma işlemi

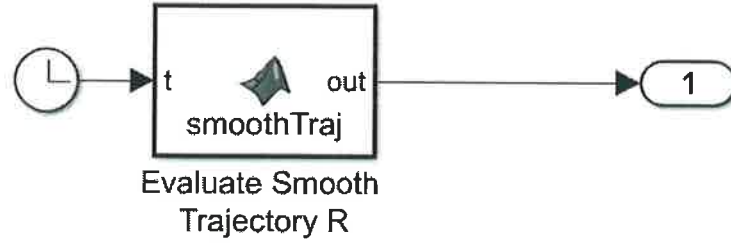
Hexapod robotun istenildiği noktaya gönderilebilmesi için bir yörünge oluşturulması gerekir. Bu yörünge tamamlanma süresi ve gidilecek olan noktaların seçilmesi bu oluşum içerisinde belirlenir. Bir önceki konuda X ekseninin sinyal girişi ile kontrol edildiğini Y ekseninin ise yörünge belirleme ile kontrol edildiğini söylemiştik. Yörünge belirleme işlemi için Matlab ortamında SmoothTrajectory komutu kullanılır. Bu komut içerisinde;

$$\text{functionout} = \text{smoothTraj}(t, \text{curveData}) \quad (5.1)$$

$$\text{out} = \text{evalSmoothTrajectory}(\text{curveData}, t); \quad (5.2)$$

$$\text{end}(\text{Anonim}) . \quad (5.3)$$

Komutu kullanılır. Burada fonksiyon çıkışı zaman ve belirlenen data olarak atanır. Daha sonra evalSmoothTrajectory çağırılır ve belirlenen fonksiyon dataları içerisine atılır ve son komutu uygulanarak çıkılır. Çağırılan Trajectory dosya komutları tez sonunda verilmiştir.



Şekil 5.70. Trajectory oluşturma

Şekil 5.70’de simulinkte kullanılan görünümü ile yörünge komutu verilmiştir. Girişine zaman değeri 1 olarak uygulanmıştır. Çıkışı 1 numaralı çıkışa verilmiştir. Bu çıkış ise Trajectory göstergesini bağlanmıştır.

Oluşturulan yörünge evalSmoothTrajectory dosyasından çağırıldığında Matlab ana ekran Workspace’in de “merkez\_motion” dosyası oluşur. Bu dosya içerisine girerek robotun Y ekseninde gideceği noktalar seçilir. Girilecek olan nokta sayısı buradan ayarlanabilir.

Variables - merkez_motion				
merkez_motion				
8x1 double				
	1	2	3	4
1	0			
2	3			
3	10			
4	13			
5	3			
6	-3			
7	-8			
8	0			
9				
10				
11				

Şekil 5.71. Yörünge oluşturma

Şekil 5.71’de merkez motion dosyası içinde Y eksenini için belirlenen yörünge noktaları gösterilmiştir. Daha sonra atanmış olan verinin içerisine yörünge oluşturma komutu olan createSmoothTrajectory komutu ile içerisine atılacak değerler ve süre tanımlanır. Bu işlem yapıldığında belirtilen nokta sayısına göre zaman aralıkları belirlenir. İşlem aşağıdaki şekilde gösterilmiştir. Örneğin robot başlangıç noktasından ikinci noktaya 28.5 saniyede gitmek zorundadır. İşlem Şekil 5.72’de gösterilmiştir.

```
Command Window
>> curveData = createSmoothTrajectory(merkez_motion,200)

curveData =

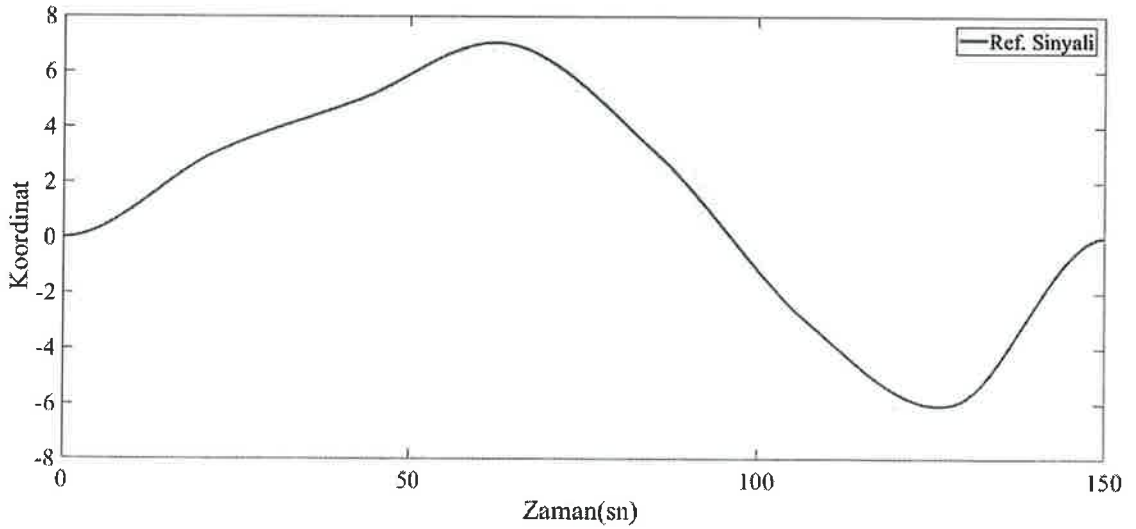
  struct with fields:

    gaitPeriod: 200
    gaitTime: [0 28.5714 57.1429 85.7143 114.2857 142.8571 171.4286 200]
    a0_merkez: [7x1 double]
    a1_merkez: [7x1 double]
    a2_merkez: [7x1 double]
    a3_merkez: [7x1 double]

>> plotSmoothTrajectory
>> |
```

Şekil 5.72. Noktalar arası zaman farkı oluşturma işlemi

Merkez motion dosyası içerisinde tanımlanan noktalar Matlab ana sayfa ekranında “plotSmoothTrajectory” komutu kullanılarak seçilen noktalar grafik üzerinde gösterilebilir. İlgili grafik Şekil 5.73’de gösterilmiştir.



Şekil 5.73. Yörünge grafiği

#### 5.2.14. Yüzey ile robot arasında sürtünme tanımlama

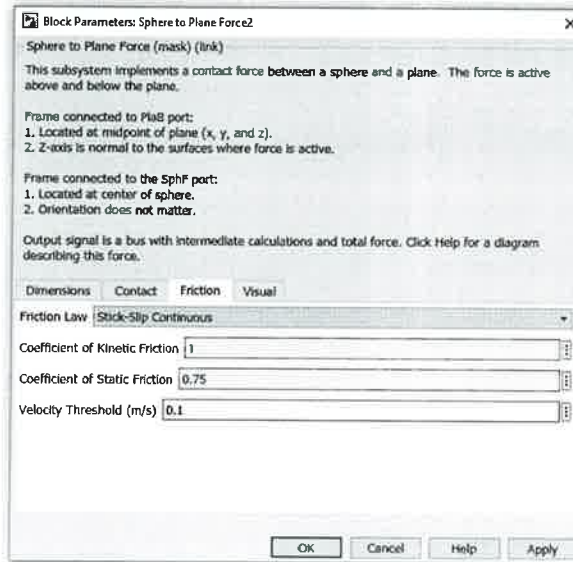
Cismin yüzey üzerindeki hareketini engelleyecek yönde yüzey tarafından uygulanan kuvvete sürtünme kuvveti denir. Sürtünme kuvvetinin yönü her zaman cismin hareketinin tersi yöndedir. Dönerek hareket eden cisimlerde sürtünmenin yönü farklıdır. Bu hareket biçiminde sürtünmenin yönü ise cismin yönü ile aynıdır (Anonim).

Sıvı sürtünme kuvveti cismin hareket ettiği yüzey ile arasında herhangi bir sıvı veya gazın bulunduğu durumdur. Kuru sürtünme kuvveti ise cismin hareket ettiği yüzey ile arasında herhangi bir madde olmadan doğrudan temas etmesi durumudur(Ferdinand, 2002).

	Malzeme	Statik Sürtünme Kat Sayıları
1.	Metal-Metal	0.15-0.60
2.	Ahşap-Metal	0.20-0.60
3.	Taş-Metal	0.30-0.70
4.	Deri-Metal	0.30-0.60
5.	Ahşap-Ahşap	0.25-0.50
6.	Deri-Ahşap	0.25-0.50
7.	Taş-Taş	0.40-0.70
8.	Taş-Kauçuk	0.60-0.90
9.	Tahta-Tahta	0.30-0.70

Tablo 5.1. Bazı cisimler arası statik sürtünme kat sayıları(Ferdinand, 2002).

Yüzey ile robot arasındaki sürtünme, hareket kontrolü için oldukça önemlidir. Bu durumda robotun ayaklarındaki pabuçlar ile uzay boşluğunda tanımlanmış olduğumuz yüzey arasındaki sürtünme katsayıları ön plana çıkmaktadır. Robotun ayaklarındaki pabuçlar kauçuk yapıdadır ve hareket zemini sert taş zemindir. Bu durumda Tablo 5.1’de verilmiş olan sekizincisatırdaki bilgi robot ve zemin arasındaki sürtünme katsayısı olarak kullanılabilir.



Şekil 5.74. Statik sürtünme tanımlama

Taş-kauçuk sürtünme aralığının tam orta noktası olan 0.75 değeri robot ile yüzey arasına sürtünme kat sayısı olarak tanımlanmıştır. İşlem Şekil 5.74'de gösterilmiştir.

Robotun yüzey ile olan sürtünme katsayısı tanımlandıktan sonra robotun üzerinde bulunan parçaların her birinin gerçek ağırlık değerleri sistemde tanımlanır. Bu tanımlama Matlab'in XML dosyasını veri dosyalarına dönüştürdüğü sıradaki oluşan veri dosyalarından değiştirilir. Robotun parça hesapları yaklaşık olarak hesaplandığında 4Kg olarak elde edilmiş ve ağırlık değiştirme işlemi Şekil 5.75'de gösterilmiştir. Her parçaya tek tek ağırlık tanımlanmamış sadece şasinin ağırlığı tüm parçaların ağırlığı olarak girilmiştir.

The image shows a software interface with two main windows. The left window is an XML editor displaying code for defining solids. The right window is a 'Solid: Solid' properties dialog box.

**XML Code (Left Window):**

```
771 -     };
772 -     smiData.Solid(13).opacity = 1;
773 -     smiData.Solid(13).ID = 'pabuc*:*Varsayilan';
774 -     smiData.Solid(14).mass = 0.07 101394187576192;
775 -     smiData.Solid(14).CoM = [1.9830103339878016 -1.95859203422
776 -     ];
777 -     smiData.Solid(14).MoI = [2.5651635750288144 2.365658945900
778 -     ];
779 -     smiData.Solid(14).PoI = [-0.0010144897742406056 0.00019263
780 -     -0.0010152158594840834];
781 -     smiData.Solid(14).color = [0.792156862745098 0.81960784313
782 -     ];
783 -     smiData.Solid(14).opacity = 1;
784 -     smiData.Solid(14).ID = 'ust_cerceve*:*Varsayilan';
785 -     smiData.Solid(15).mass = 4.0;
786 -     smiData.Solid(15).CoM = [6.889129048804131 -3.047354883659
787 -     ];
788 -     smiData.Solid(15).MoI = [2.0789459309905367 2.052022917965
789 -     ];
790 -     smiData.Solid(15).PoI = [5.0555617316970664E-9 1.055125291
791 -     ];
792 -     smiData.Solid(15).color = [0.792156862745098 0.81960784313
```

**Solid: Solid Properties Dialog (Right Window):**

Description: Represents a solid combining a geometry, an inertia and mass, a graphics component, and rigidly attached frames into a single unit. A solid is the common building block of rigid bodies. The Solid block obtains the inertia from the geometry and density, from the geometry and mass, or from an inertia tensor that you specify.

In the expandable nodes under Properties, select the types of geometry, inertia, graphic features, and frames that you want and their parameterizations.

Port R is a frame port that represents a reference frame associated with the geometry. Each additional created frame generates another frame port.

Properties:

Property	Value	Unit
File type	STEP	
File Name	ustcerceve_Varsayilan_sldprt.STEP	
Type	Custom	
Mass	smiData.Solid(15).mass	kg
Center of Mass	smiData.Solid(15).CoM	cm
Moments of Inertia	smiData.Solid(15).MoI	kg*cm^2
Products of Inertia	smiData.Solid(15).Pol	kg*cm^2

Buttons: OK, Cancel, Help, Apply

Şekil 5.75. Oluşturulan XML dosya verilerinde kütle değişimi

Robotun ilk kontrol analizleri yapıldığı esnada ağırlığı çok küçük olduğu için yüzey tepkisi robotu karşılayabilecek düzeydedir. Fakat robotun ağırlığı gerçek parça ağırlıkları ile değiştirildiğinde yüzey robotu taşıyamayacaktır ve robot yüzeyi delerek uzay boşluğuna düşecektir. Simulink üzerinde yaşanan bu olay Şekil 5.76'da ayrıntılı olarak gösterilmiştir.



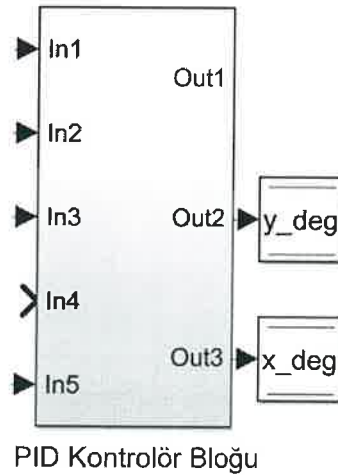
Şekil 5.76.Robotun yüzeyde tutunamaması

Robotun yüzeyi delerek boşluğa düşmesini engellemek için yüzey tepki kuvvetini artırmak gerekir. Bilindiği üzere tepki kuvveti cismin ağırlığına eşit veya daha yüksek olmalıdır. Aksi takdirde de yüzey cismi durduramaz ve tahrip olur. Bu olumsuz etkiyi engellemek için yüzeyin tepki kuvveti robotun ağırlığına eşitlenmiştir.

### 5.2.15. PID kontrol

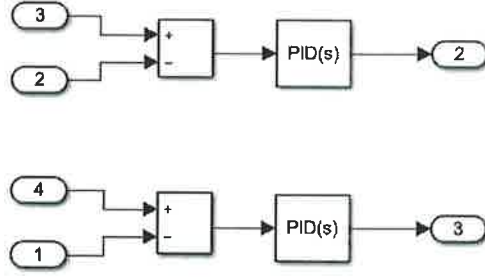
Tezde uygulanan PID kontrol çalışmasında robot X ve Y olmak üzere iki eksende PID kontrolör ile kontrol edilmiştir. X eksen, sinyal bloğu ile düz bir sinyal üzerine kontrol edilmiştir. Y eksen ise kişinin kendisi tarafından belirlenen noktalardan oluşan yörünge üzerinde kontrol edilmiştir.

Kontrole başlarken ilk olarak bir PID kontrolör bloğu oluşturulur. Bu blok içerisine PID bloğu çağırılır. X ve Y olmak üzere iki eksende kontrol yapılacağı için blok iki defa çağırılır. Oluşturulan kontrolör bloğu Şekil 5.77’de gösterilmiştir.



Şekil 5.77. PID kontrolör bloğu

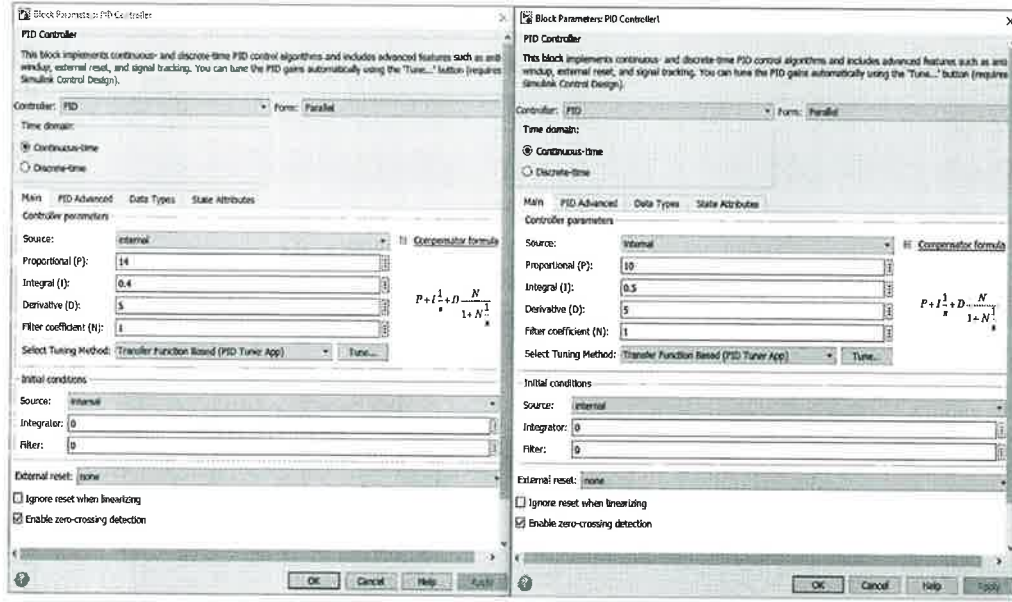
Aşağıda bloğun içerisi gösterilmiştir. In2 girişinden robotun Y koordinatındaki konumunun sisteme girişi yapılır. Bu girişin bağlandığı PID kontrolör Y eksenini kontrol eden kontrolör olarak seçilmiş olur.In3 girişinden oluşturulan yörünge değerlerinin sisteme girişi yapılır. Yörünge kontrolü Y ekseninde yapılacağı için bu değer de Y eksenini kontrolörüne bağlanır. Bağlantısı yapılan bu iki değer farkı alınır. Aradaki fark büyük ise PID kontrolör daha büyük değerler üreterek farkı kapatmaya çalışır ve robotun ayaklarını açıp kapama açıları ve hızı buna göre değişir.



Şekil 5.78. PID kontrolör bloğunun iç görünümü

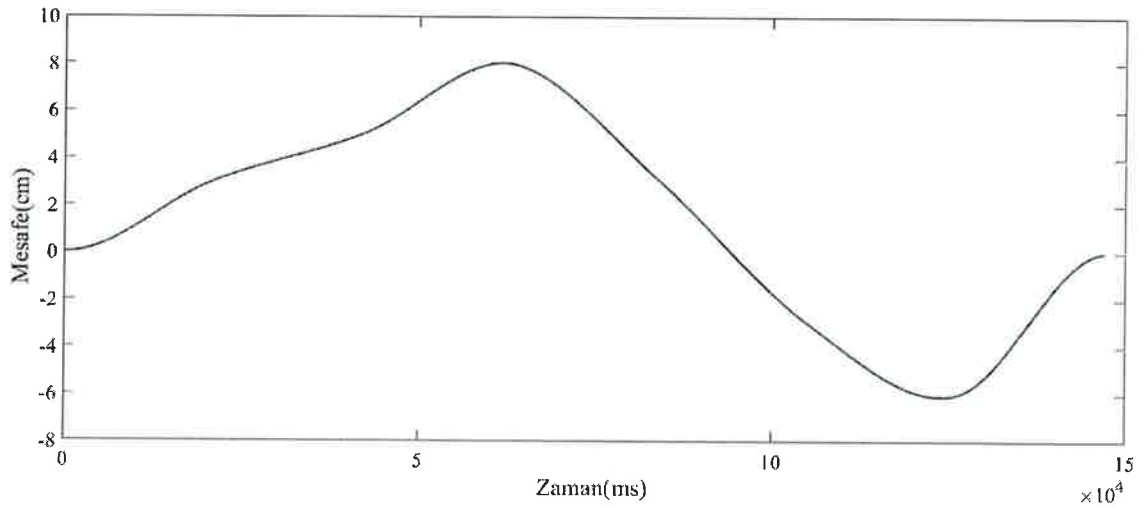
In1 girişinden robotun gövdesinin X eksenini üzerindeki konumunun sisteme girişi yapılır. Bu giriş blok içerisinde bulunan ikinci PID kontrolöre bağlanır. İkinci kontrolörde X eksenini kontrolörü olarak seçilmiş olur. In5 girişinden ise sinyal giriş bloğu oluşturma başlığından oluşturulan bloktan gelen sinyal girişi alınır ve X eksenini kontrolörüne bağlanır. Bağlantısı yapılan iki girişin farkı alınır. Farka bağlı olarak PID kontrolör değer üretir ve X eksenini üzerinde referansı yakalamaya çalışır. PID kontrolör çıkışları kendi atamış olduğumuz `zy_deg` ve `x_deg` isimli veri depolarına gönderilir. Bu depolar doğrudan çağırılarak robot bacak eklemlerinde bulunan servo girişlerine bağlanır. Çıkışların doğrudan girişe bağlanmamasının sebebi komut ekledikçe programın ağır çalışması ve hata vermesinden kaynaklıdır.

PID kontrolörün içerisinde barındırdığı P,I ve D parametlerini ayrıntılı anlatmıştık. Tezimizde kontrolörlerin içerisinde bulunan değerler Şekil 5.79'da gösterilmiştir.



Şekil 5.79. PID kontrolör parametre değerleri

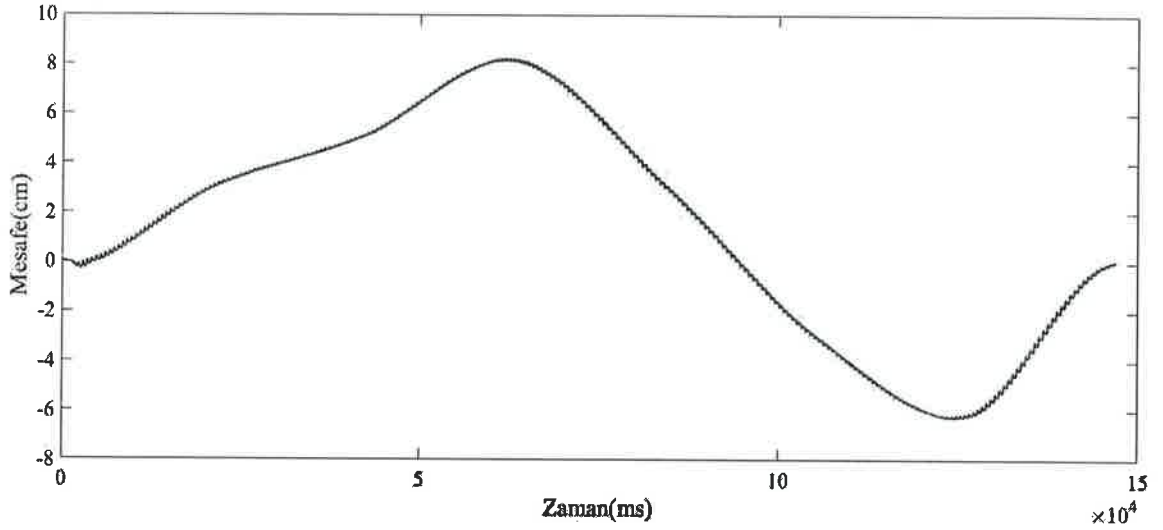
Bu değerlerden P değeri robotun takip etmesi gereken referans sinyali ile konumu arasındaki hata değerini kontrol eder. Bu değer X eksen için 14 ve Y eksen için 10 olarak girilmiştir. Sistemin hata süresini kontrol eden I değeri ise X eksenin 0.4 olarak Y ekseninde ise 0.5 olarak girilmiştir. Hata değişimini kontrol eden D değeri ise iki ekseninde de 5 olarak girilmiştir. Kontrolörün kendi içerisinde bulunan filtre kat sayıları 1 olarak ayarlanmıştır. Sistem girilen bu değerlerde çalıştırılmıştır. Değerleri elde etme sırasında Matlab'ın kat sayı belirlemek için kullanılan "Tune" katsayı fonksiyonu servo motorların fazla olmasından dolayı kullanılamamıştır. Değerler bir diğer yöntem olan deneme yanılma yolu ile elde edilmiştir.



Grafik 5.1. Y eksen takibinde izlenecek olan referans sinyali

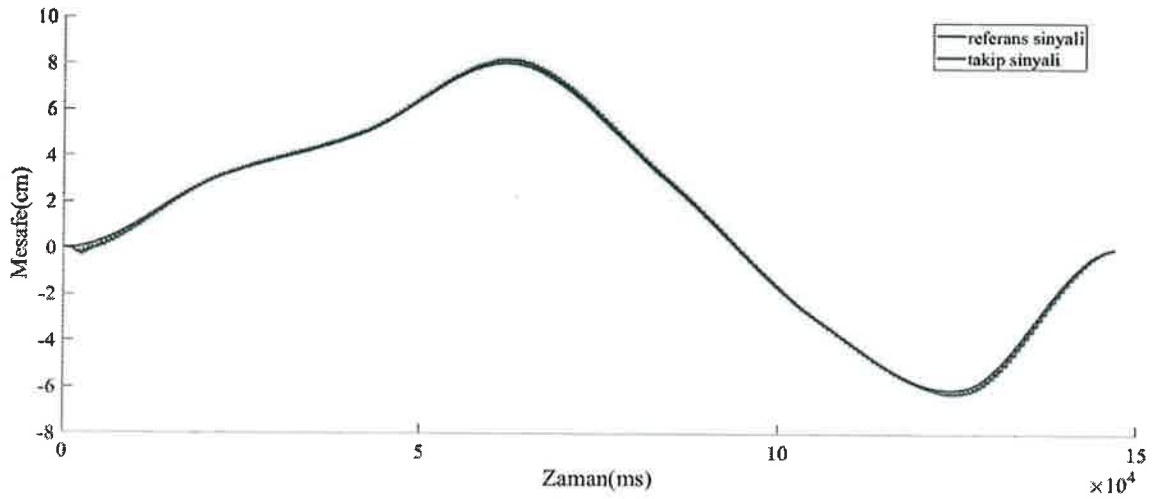


Robot y ekseninde Trajectory kontrol ile x ekseninde ise sinyal bloğu ile kontrol edilmiştir. Grafik 5.1’de görülen sinyalde robotun y ekseninde izleyeceği referans sinyali görülmektedir.



Grafik 5.2. Y ekseninde referans yörüngeyi takip sinyali

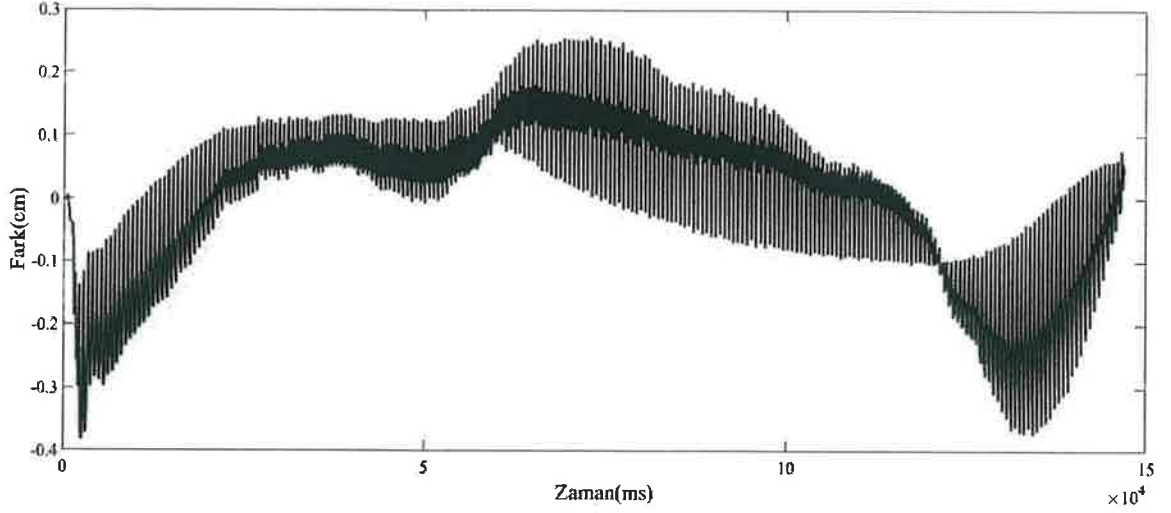
Y ekseninde robotun PID kontrolör ile referans sinyalini takip etmesi sonucunda Grafik 5.2 elde edilmiştir. Grafikte robotun hareket süresinin tamamı alınmıştır. Başlangıçta bir salınım görülmektedir. Bu salınım robotun referans sinyaline yetişmesi sürecinde oluşan hatadır.



Grafik 5.3. Y eksenindeki referans ve takip sinyali

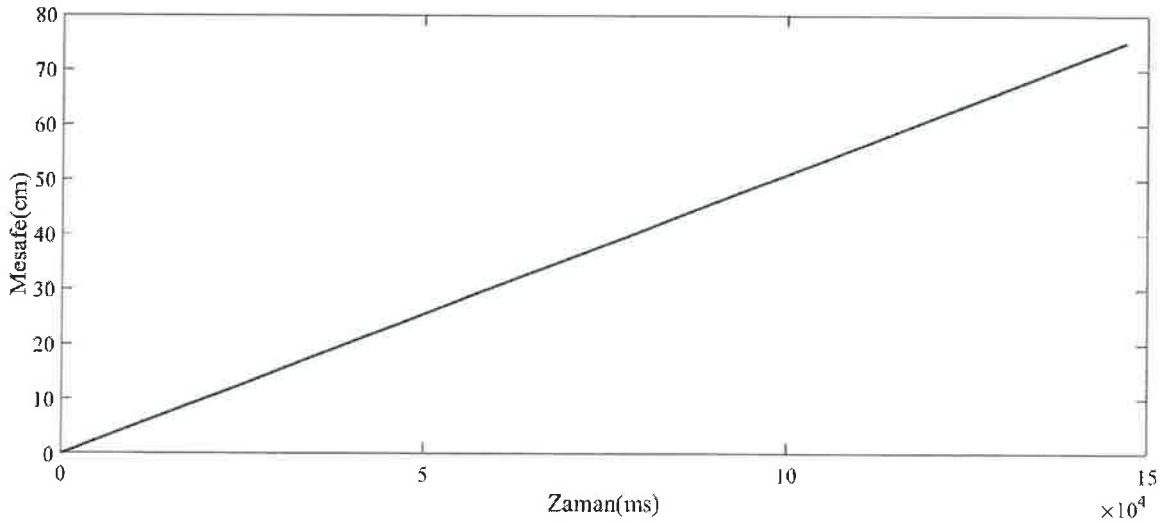
Y ekseninde kontrol sağlamak için oluşturulan referans sinyali ile robotun y ekseninde yapmış olduğu takip sinyalini Grafik 5.3’de gösterilmiştir. Yörüngeli takip yönteminde önemli husus dönüş noktalarında (tepelerde) hızlı dönüş sağlayabilmek ve dönüş

noktalar dışında sinyallerin üst üste gelebilmesidir. Grafikte görüldüğü gibi Y ekseninde referans sinyali takibi robot tarafından başarılı bir şekilde gerçekleştirilmiştir.



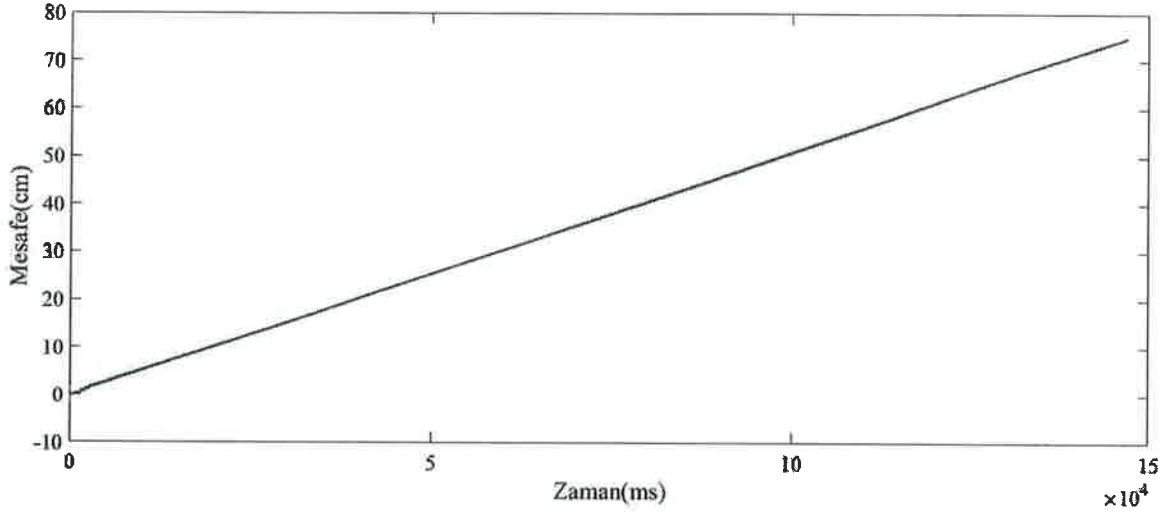
Grafik 5.4. Y eksenindeki referans ve takip sinyali hatası

Y eksenini takibinde oluşturulan yörünge sinyali ve robotun yörüngeyi takip etmesi sonucu oluşan y eksenini takip sinyalinin farkı alınarak elde edilen hata değerleri Grafik 5.4'de gösterilmiştir. Tüm hareket zamanı dikkate alınan grafikte, başlangıçta robotun referans sinyalini yakalama sürecinde oluşturduğu hata değeri görülmektedir. Bu hata tüm kontrol yöntemlerinde görülmektedir. Bu hatanın dışında kalan değerler 0.1cm ve -0.1cm aralığında olduğu görülmektedir. Bu aralık ihmal edilebilir aralıktır ve takibin başarılı bir takip olduğu söylenebilir. Yorumlanan grafik kabaca Grafik 5.3'e benzetilebilir.



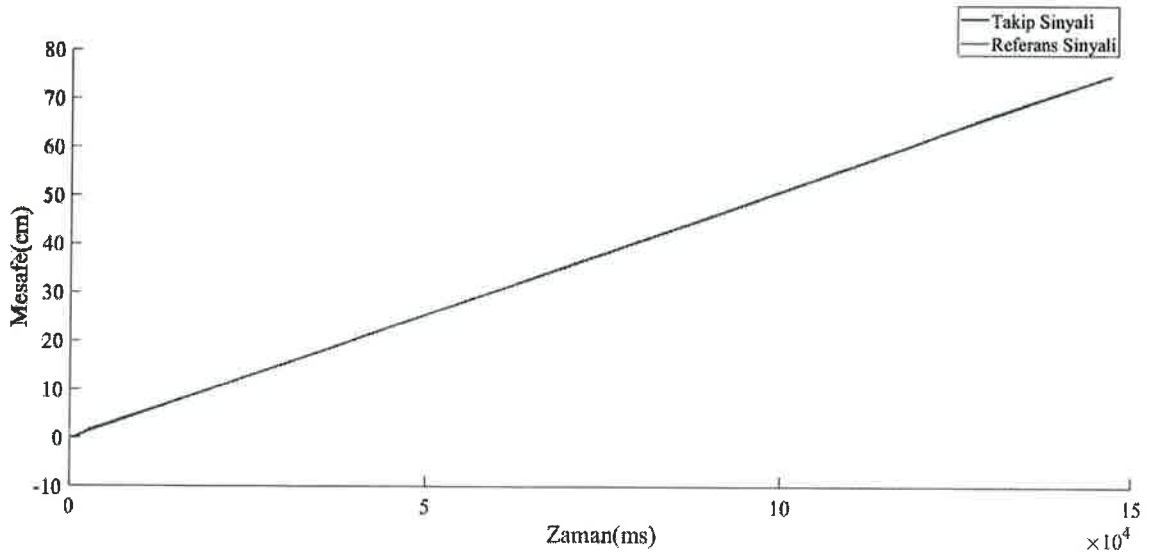
Grafik 5.5. X ekseninde takip edilecek olan referans sinyali

Grafik 5.5’de görülen sinyal, sinyal oluřturma bloęu ile oluřturulmuř olan referans sinyalidir. Bu sinyal X ekseninde takip edilecek olan sinyaldir. X eksenindeki ilerleme doęrusal döz bir ilerleme olup süre ile bir baęlantısı yoktur.



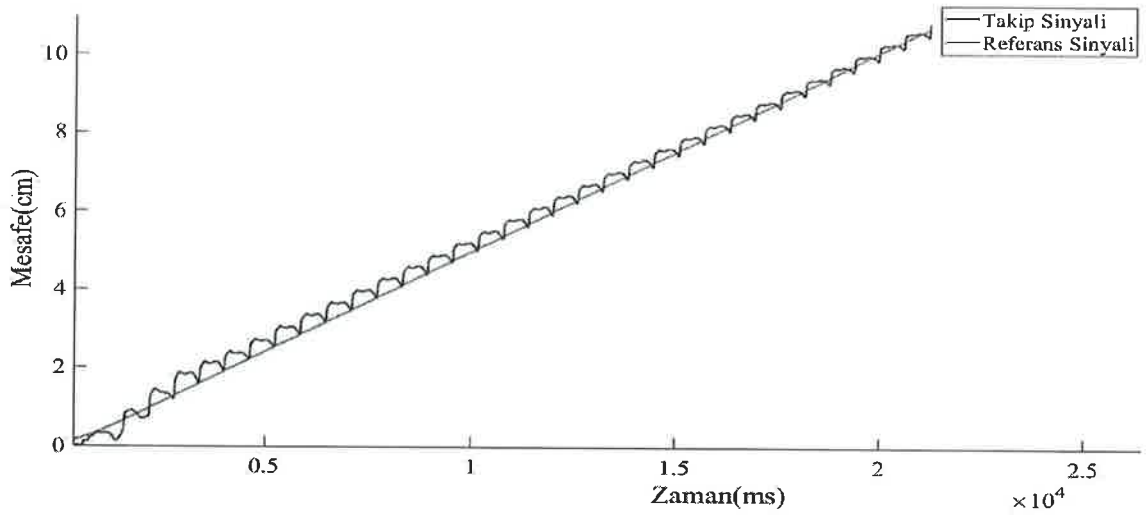
Grafik 5.6. X eksenindeki takip sinyali

Sinyal bloęu tarafından üretilen sinyal referansının PID kontrolör ile takibi Grafik 5.6’da verilmiřtir. Doęrusal bir sinyal olduęu için robot sinyali 1ms gibi kısa bir sürede yakalamıř ve yakaladıęı andan hareketin bittięi ana kadar çok iyi bir řekilde takip etmiřtir.



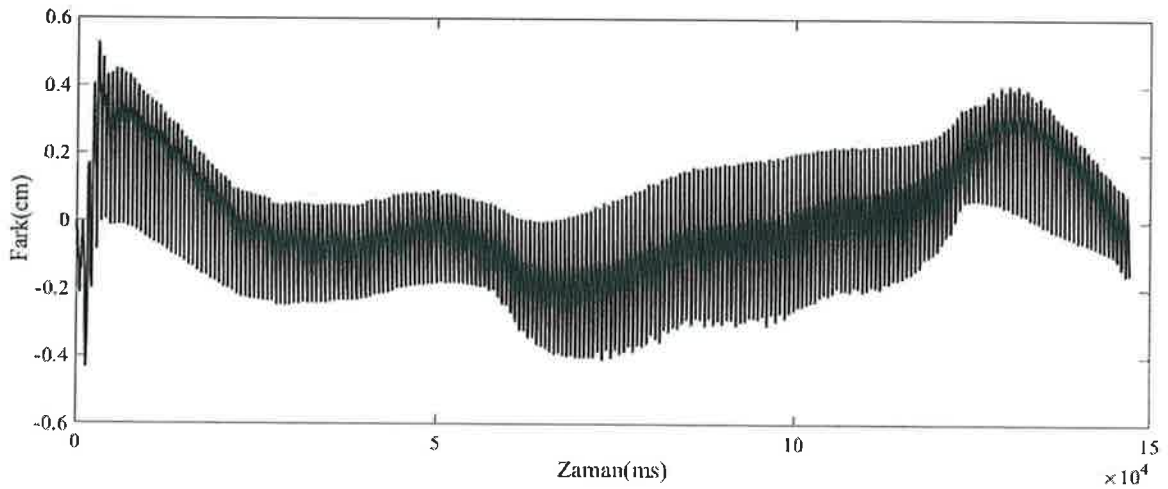
Grafik 5.7. X eksenindeki referans ve takip sinyali

Referans sinyali ile takip sinyalinin görüldüęü üstteki grafikte sinyaller arası belirgin bir fark olmadıęı görölmektedir. Hata deęerleri düşük olduęu için takip oldukça başarılıdır.



Grafik 5.8.X eksenindeki referans ve takip sinyallerinin 0-2,3ms aralığı

Takipte hata çok minimal olduğu için daha ayrıntılı görebilmek amacı ile zamanda 0-2.3 ms aralığı Grafik 5.8'de gösterilmiştir. Şekilde, robotun bahsi geçen sürenin sonuna doğru sinyali ortaladığı görülmektedir. Takipteki dalgalı yapının sebebi robotun Wave(dalga) yürüyüşü ile kontrol edilmiş olmasıdır.

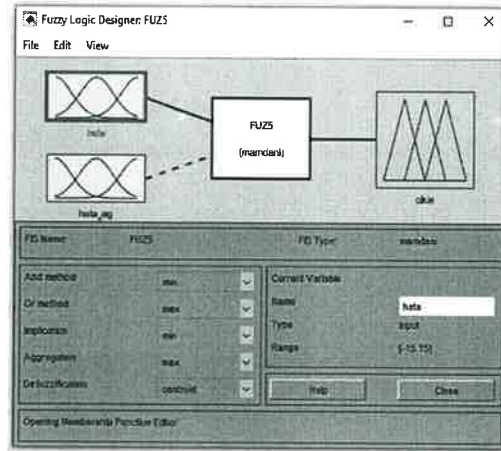


Grafik 5.9.PID kontrolde X eksenindeki hata

Robotun PID kontrolör ile x ekseninde kontrol edilmesi aşamasında, referans sinyali ve takip sinyalinin farkı alınarak oluşturulan Grafik 5.9'da hareketin tamamında ki hata görülmektedir. Hareketin tamamen gösterildiği grafikte hatanın 5mm ve -5 mm aralığında olduğu görülmektedir. Bu grafik analiz edildiğinde robotun x ekseninde takibi çok küçük ihmal edilebilir hatalar ile tamamladığı görülür.

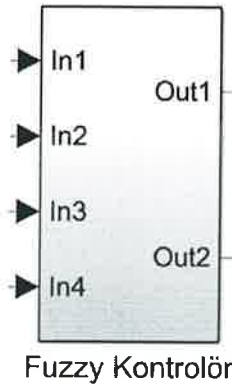
## 5.2.16. Fuzzy kontrol

Fuzzy kontrol sayfasını açmak için komut sayfasına “Fuzzy” yazılır. Açılan pencerede “edit” kısmından sistemde kullanılacak olan giriş ve çıkışlar eklenir. Sırası ile eklenen girişlere çift tıklanır. Giriş içerisinde bulunan her bir dalgaya isim verilir ve girişin hata aralığı belirlenir. Girişler tanımlandıktan sonra çıkışlara girilir ve çıkışlar içinde tepe noktalar isimlendirilerek hata aralığı belirlenir. Şekil 5.80’de Fuzzy giriş ekranı gösterilmiştir.

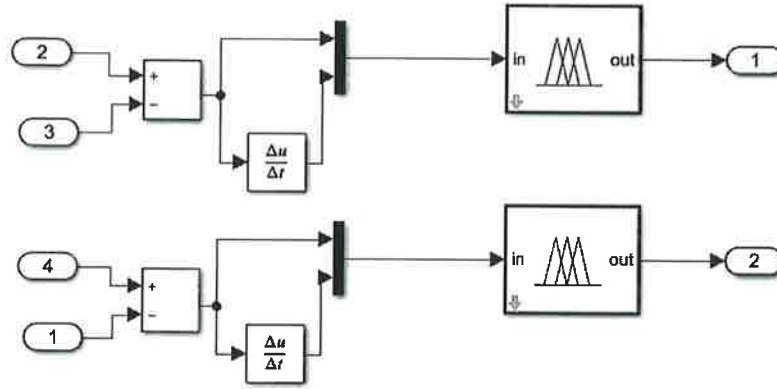


Şekil 5.80. Fuzzy blok tasarımı

Yukarıdaki şekilde tezde kullanılmak için oluşturulan Fuzzy kontrolör blok görülmektedir. Şekilde de görüldüğü gibi 2 tane giriş ve bir tane çıkış değeri vardır. Tıpkı PID kontrolörde olduğu gibi Fuzzy kontrolde de her eksen için bir Fuzzykomutu kullanılır. Sistemde kullanılmak üzere oluşturulan Fuzzy Kontrolör bloğu aşağıdaki şekilde gösterilmiştir.



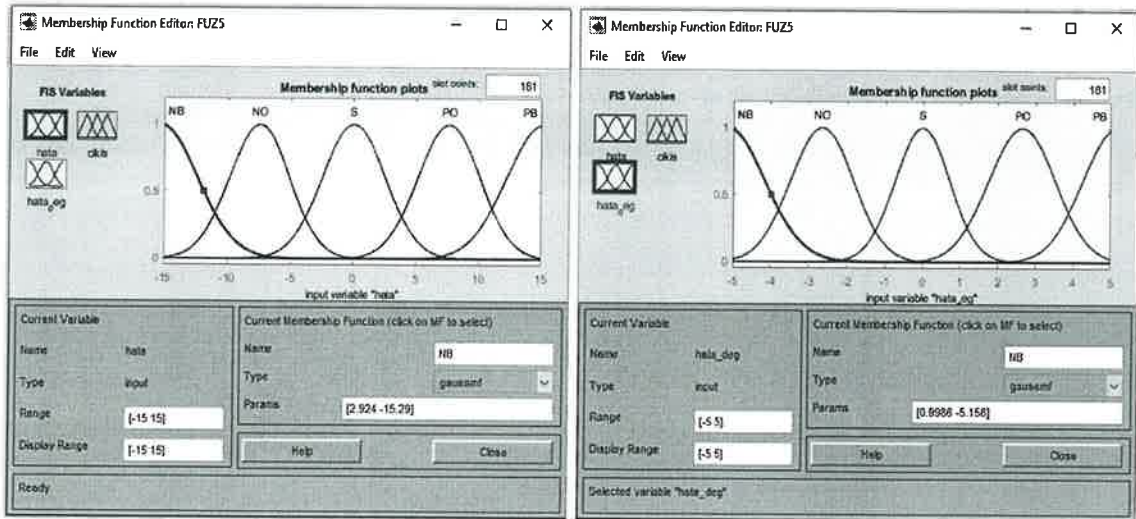
Şekil 5.81. Fuzzysimulink bloğu



Şekil 5.82. Fuzzy simulink bloğu içeriği

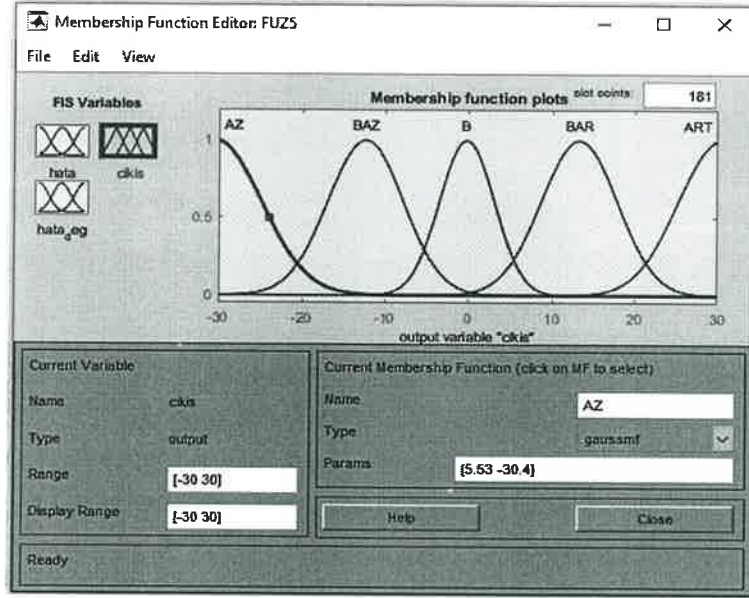
Şekil 5.82'de Fuzzy bloğunun içeriği görülmektedir. PID Kontrol tipinde işlediğimiz gibi giriş sinyalleri Fuzzy Kontrol içinde aynıdır. Yukarıda ki sistem Y eksenini için ve diğer sistem ise X eksenini için kurulmuştur. Bu sistemlere gelen girişler, robotun anlık X-Y koordinatı ile robotun izlemesini istediğimiz referans girişleridir. Bu sinyallerin farkı alınır. Sinyallerin farkının alındığı çıkış 2 yola ayrılır bu yollardan birisi doğrudan diğeri ise türevi alınarak Fuzzy bloğuna giriş yapılır. Fuzzy blok çıkışı ise robotun ayak eklemlerine girilir.

Fuzzy blokları içerisinde tasarlanan FLD'ler isimlendirilir. Sistem çalıştırıldığında programdaki Fuzzy Bloğu otomatik olarak tasarlanmış olan FLD'yi çağırır ve çalıştırır. FLD'lerin tasarlanması ise kontrolün uygulanacağı alana uygun kurallar oluşturularak yapılır.



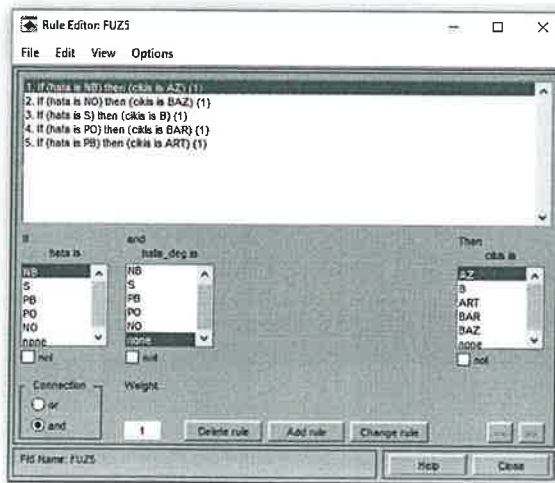
Şekil 5.83. Fuzzy girişleri

Şekil 5.83’de kullanılacak olan Fuzzy sistem için tanımlanan iki giriş görülmektedir. Bu girişlerin tepe noktaları ve hata aralıkları kural yazarken kullanılmak üzere isimlendirilmiş ve belirlenmiştir.



Şekil 5.84. Fuzzy çıkışı

Şekil 5.84’de sistem için tanımlanan çıkış ve hata aralıkları için verilen isimler ve aralığı değeri görülmektedir. Bu hata aralıkları sistemin kararlı çalışması için önemlidir. Eğer sistemin hatası az ise ve yüksek hata aralığı girilmiş ise sistemin hassasiyeti azalır. Bu durumda aralığın azaltılması gerekir. Böylece sistemin daha hassas çalıştığı gözlemlenir. Eğer küçük hata aralığında yüksek hatalar alınıyor ise buda sistemi olumsuz etkiler.



Şekil 5.85. Fuzzy kontrol için yazılan kurallar

Şekil 5.85’de Fuzzy kontrolde kullanılacak şekiller gösterilmiştir. Kural satırında yazılan kuralların açıklaması:

Satır 1- Eğer hata Negatif yönde ve Büyük ise(NB) çıkış değerini AZALT(AZ)

Satır 2- Eğer hata Negatif yönde ve Orta büyüklükte ise(NO) çıkış değerini Biraz Azalt(BAZ)

Satır 3- Eğer hata Sıfır ise(S) çıkış değerini koru

Satır 4- Eğer hata Pozitif yönde ve Orta büyüklükte ise(PO) çıkış değerini Biraz Azalt(BAR)

Satır 5- Eğer hata Pozitif yönde ve Büyük ise(PB) çıkış değerini Artır(ART)

olarak verilmektedir. Bu kurallara göre Fuzzy Bulanık Mantık ile robotun referans yörüngesine olan uzaklığı kontrol edilmiştir. Kurallarda açık bir şekilde kontrol anlatılmıştır. Bu anlatılan büyüklüklere göre robotun ayaklarındaki açılar kontrol edilmektedir. Bu kontrolün yönü negatif veya pozitif olmaktadır.

`curveData = createSmoothTrajectory(merkez_motion,200)` komutunu komut sayfasına girdiğimizde oluşturulan yörüngeyi 200t zamanda tamamla bilgisi robota verilmiş olur.

Sistemin tamamlanması için verilen toplan süre(t), sisteme tanımlanmış olan nokta sayısına bölünerek bir noktadan en yakın diğer bir noktaya ulaşım için robota verilen maksimum süre elde edilir;

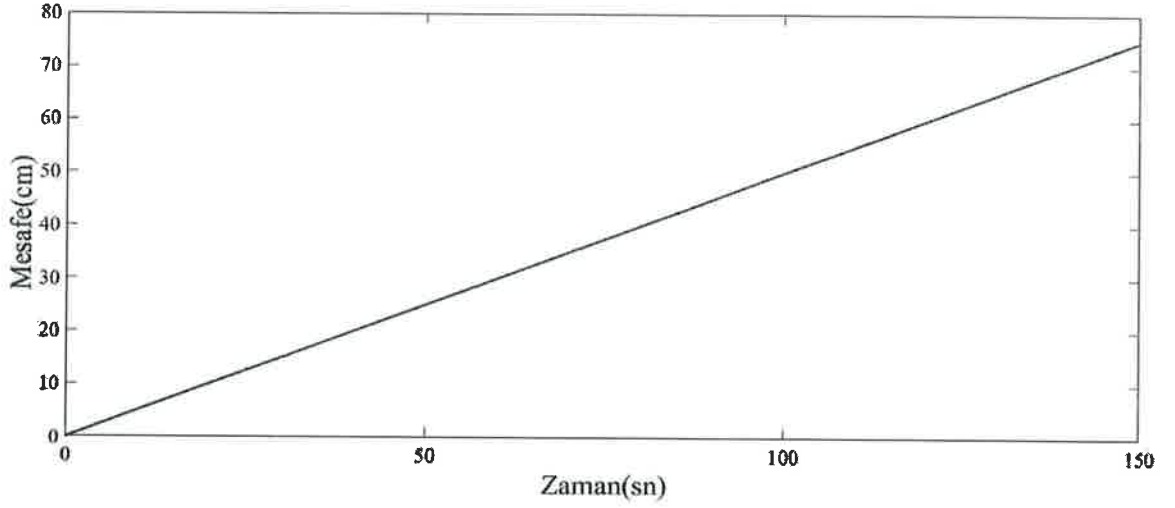
Sistemin t’si 200sn olduğundan ve sisteme girilen nokta sayısı 8 olarak girildiğinden;

$T=200/8= 25\text{sn}$  olarak bulunur.

Her bir nokta arasında robot için verilen maksimum süre 25sn olarak bulunur. Robot noktalar arası hareketinde verilen sürede yetişemez ise bacak açıları maksimum derecede hareket eder ve verilen süre içerisinde yolunu tamamlamaya çalışır.

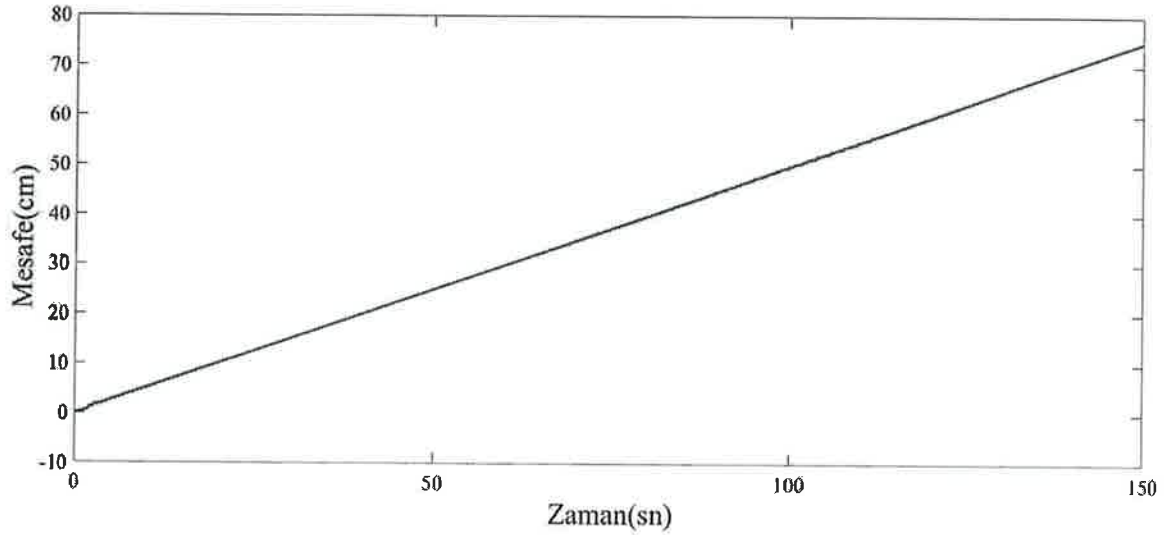
Referans ile takip sinyali arasındaki çıkış hatası, pozitif değer için 28 ve negatif hata değeri için -28 olarak girilmiştir.





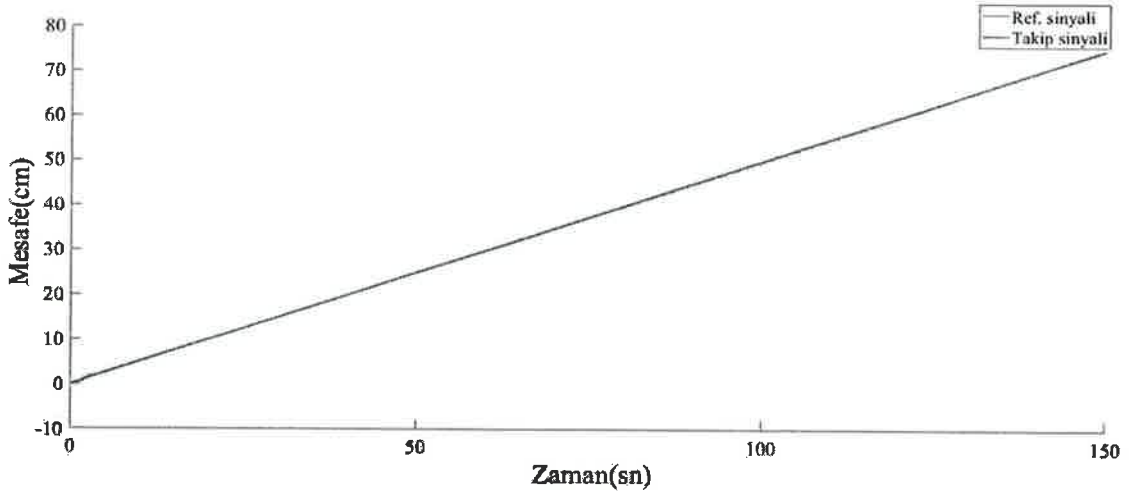
Grafik 5.10. X ekseninde takip edilecek olan referans sinyali

Grafik 5.10'da Fuzzy kontrol tipinde robotun izleyeceği X eksen referans sinyali gösterilmiştir. PID kontrol tipinde olduğu gibi süre 150sn olarak belirlenmiştir. Mesafe 80cm olarak belirlenmiştir.



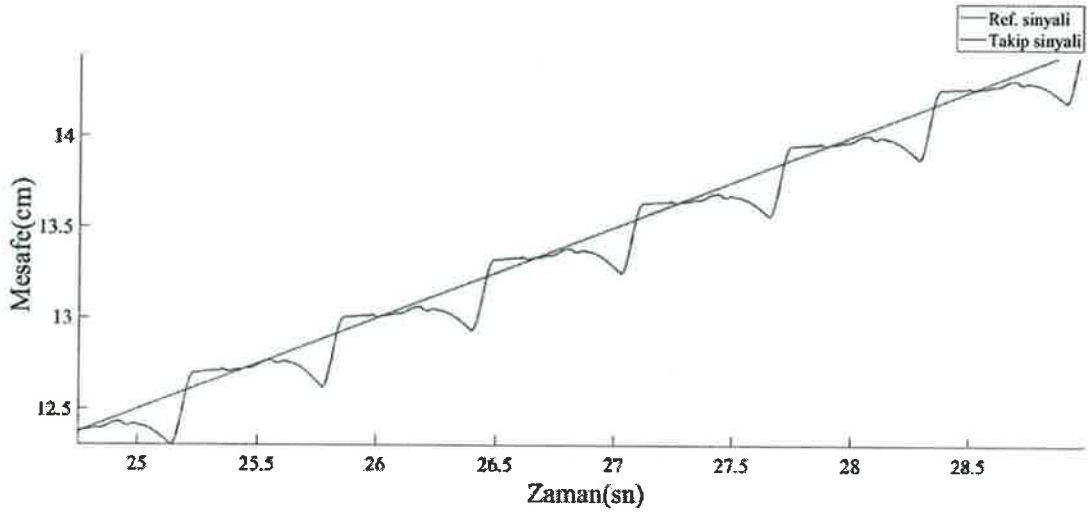
Grafik 5.11. X eksenindeki takip sinyali

Grafik 5.11'de robotun X ekseninde referans sinyalini takibi sonucu elde edilen sinyal gösterilmiştir. Başlangıçta çok küçük bir salınım gözlemlenmiştir. Bu salınım robotun referans çizgisini algılayıp yakalaması için geçen sürede oluşan hatadır. Çok küçük olduğu için ihmal edilmiştir. Devam eden sürede iyi bir takip yapıldığı gözlemlenmiştir.



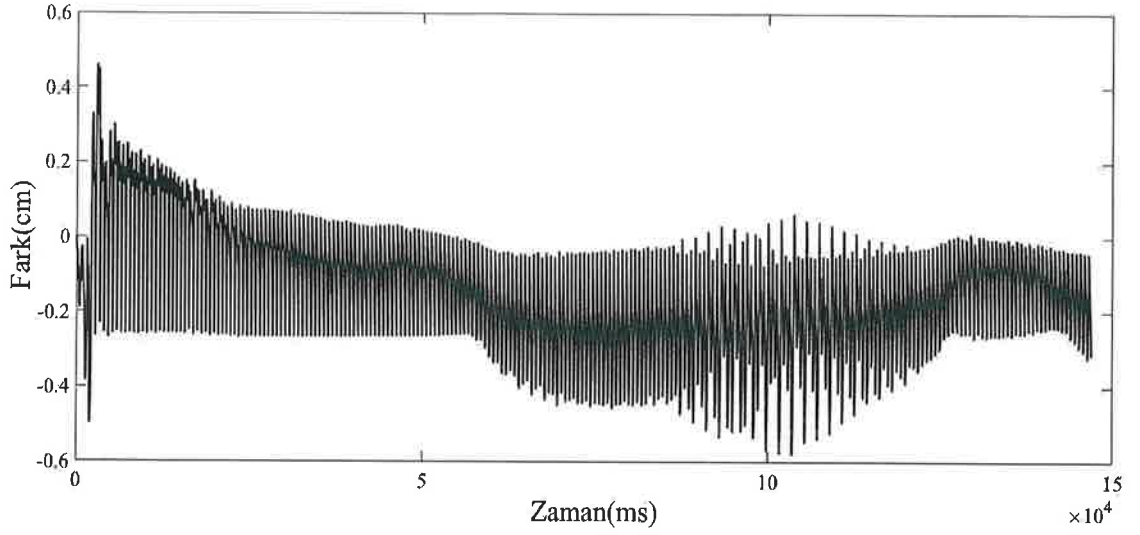
Grafik 5.12. X eksenindeki referans ve takip sinyali

X eksenindeki referans ve takip sinyallerinin ikisinin de bulunduğu Grafik 5.12'de takibi yorumlamak daha kolaydır. Neredeyse bire bir kontrol sağlanmıştır. PID kontrolör ile yapılan kontrole yakın bir takip grafiği elde edilmiştir.



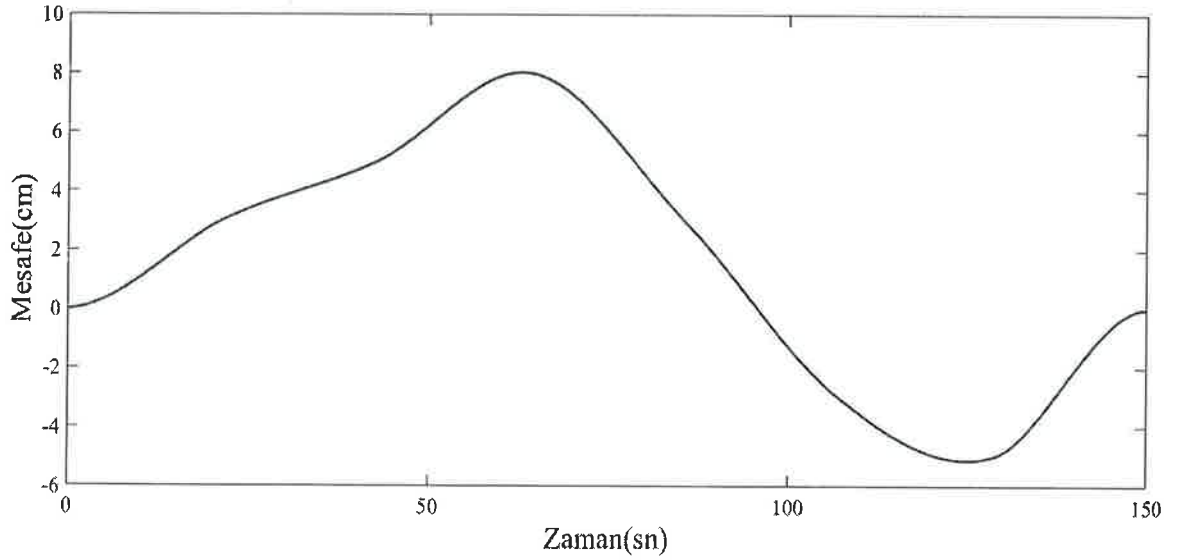
Grafik 5.13. X eksenindeki referans-takip sinyalinin 25-29sn aralığı

Robotun X eksenindeki hareketinin kontrolü sırasında 25-29sn aralığı yakınlaştırılmış gösterimi Grafik 5.13'de gösterilmiştir. Bu grafikten robotun referansı takibi daha net görülmektedir. Takip sinyalinin dalgalı olmasının sebebi robotun yürüyüş hareketlerinden kaynaklanmaktadır. Bu dalgalı yürüyüş biçimi literatüre WaveGait(Dalgalı yürüyüş) olarak geçmiştir.



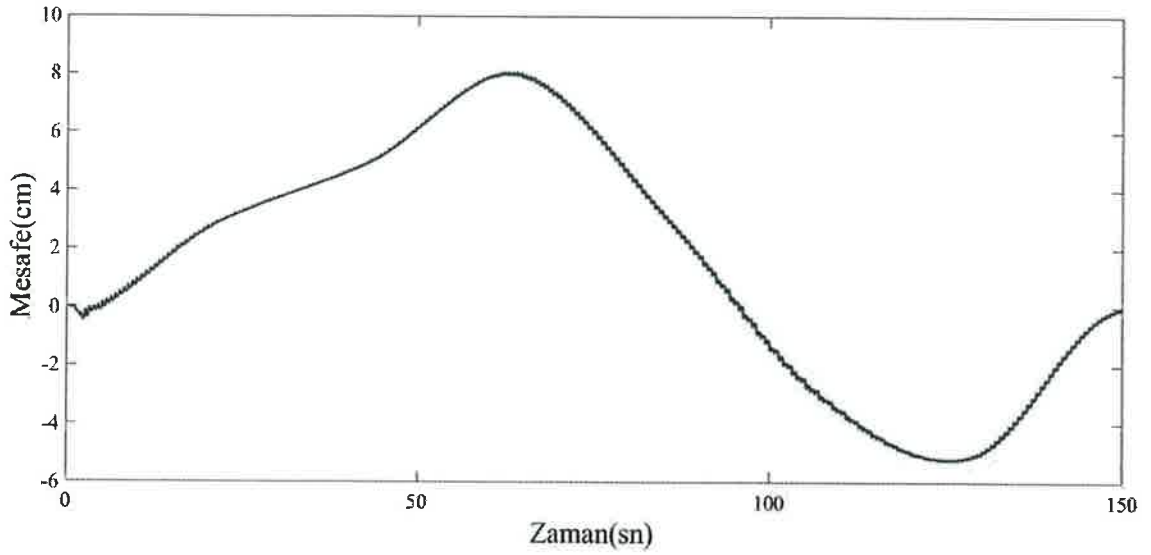
Grafik 5.14. Fuzzy kontrolde X eksenindeki hata

Grafik 5.14'de Fuzzy kontrolör ile X eksenini referans takibi işlemi robot tarafından yapılan hata sinyali görülmektedir. Hata sinyali referans ve takip sinyallerinin farkı alınarak elde edilmiştir. Tüm hareket zamanının temel alındığı grafikte hatanın 2mm ve -2mm aralığında olduğu görülmektedir. Hata değerleri arazide hareket edecek bir robot için çok küçük değerlerdedir. Hata sinyalinden başarılı bir takip yapıldığı görülmektedir.



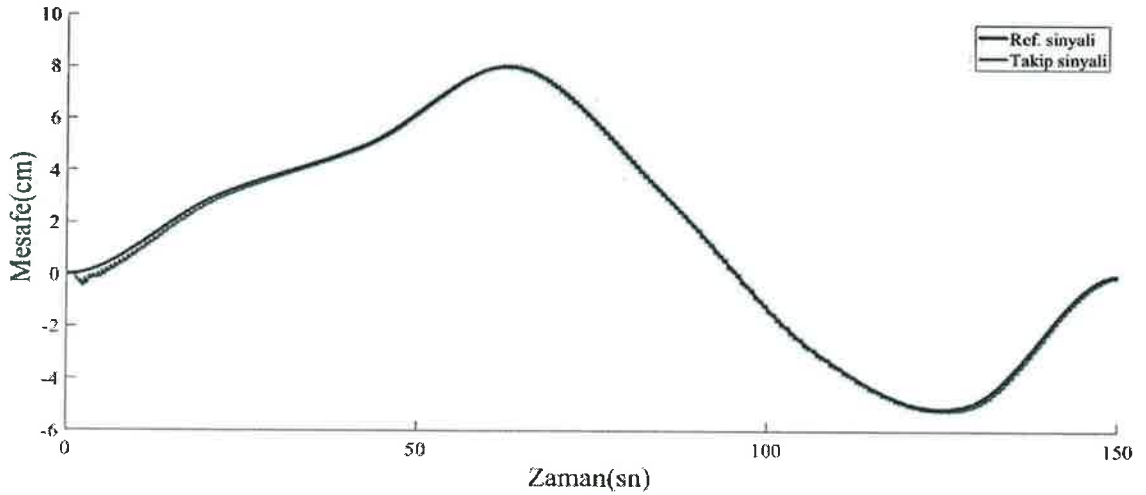
Grafik 5.15. Y eksenini için Trajectory ile oluşturulan takip sinyali

Grafik 5.15'de görülen referans sinyali yörünge oluşturma yöntemi ile oluşturulmuştur. Noktalar (0-3-5-8-3-(-3)-(-5)-0) olarak belirlenmiştir. Oluşturulan bu rota PID kontroldeki rota ile aynıdır.



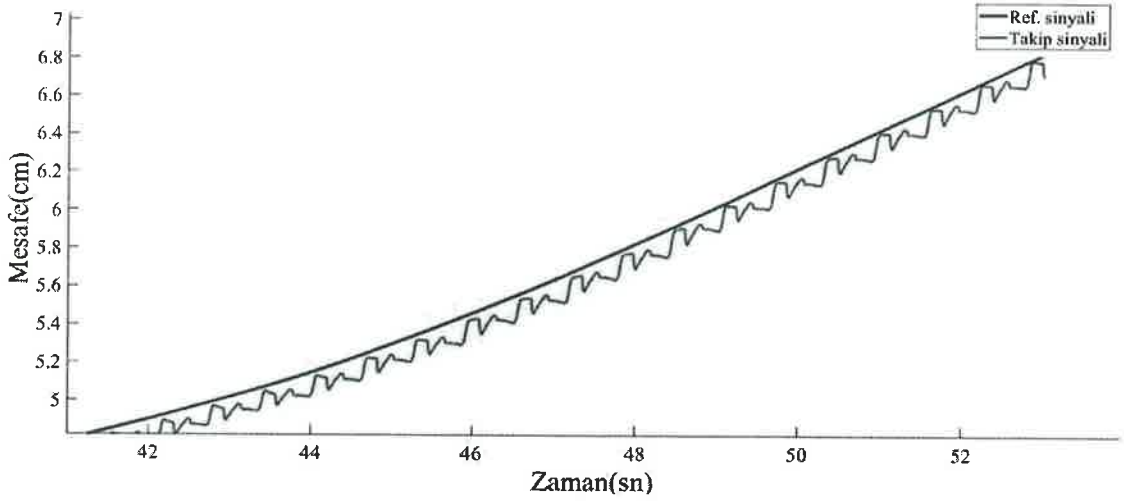
Grafik 5.16 Y eksenindeki takip sinyali

Y eksenini için verilen referans sinyalinin Fuzzy kontrolör ile robot tarafından yapılan takip sinyali Grafik 5.16'da verilmiştir. Takibin başlangıcında referans sinyalini yakalamak için geçen sürede salınım görülmektedir. Bu salınım ihmal edilebilecek seviyede küçüktür. Zamanda 80-120sn aralığında sinyalin çok küçük aralıklarla salınımlı olduğu gözlemlenmiştir. Bu kısımlarda robot ayaklarını daha büyük açılarda açarak daha hızlı hareket ederek sinyal ile arasındaki mesafeyi korumuştur.



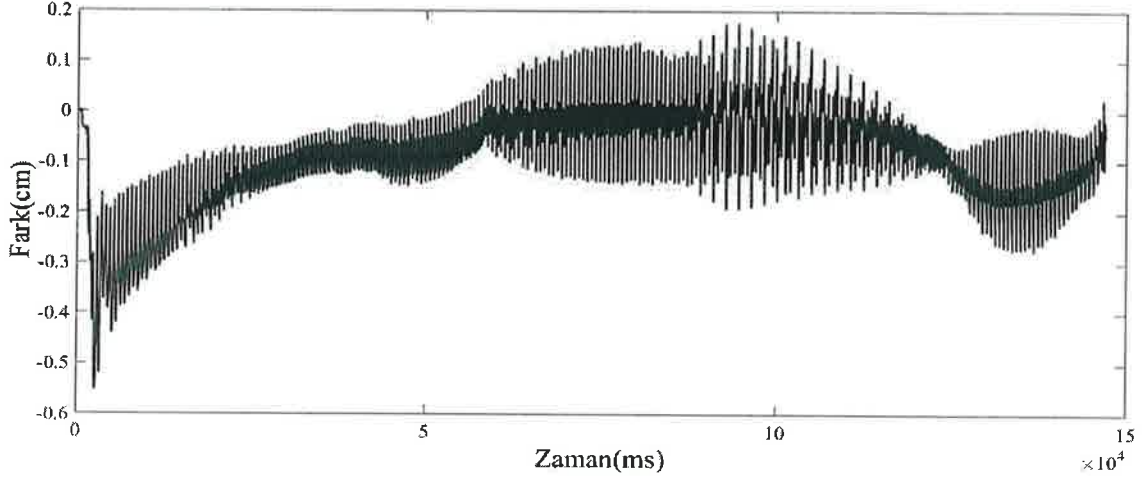
Grafik 5.17. Y ekseninde yapılan referans ve takip sinyalleri

Grafik 5.17'de Y ekseninde Fuzzy kontrol sonucu elde edilen referans ve takip grafikleri görülmektedir. Takip başlangıcından yaklaşık 10 saniye sonra robot referans sinyalini yakalayıp otonom olarak referans sinyalini takip edebilmiştir. Yörünge boyunca kabul edilebilir yörünge takip hatası elde edilmiştir. Grafik 5.17'de hata miktarını daha iyi görebilmek için Grafik 5.18 verilmiştir.



Grafik 5.18. Y ekseninde yapılan referans ve takip sinyallerinin 42-52sn aralığı

Fuzzy kontrolör ile y ekseninde referans sinyali takibi grafiğinin zamanda 42-52snve mesafede 5-7 cm aralığı Grafik 5.18’de gösterilmiştir. Robotun sinyali ortalamak için sürekli sinyale yaklaştığı gözlemlenmiştir. Bu grafikte referans yörünge eğrisi ile takip sinyali arasındaki ortalama hata miktarı yaklaşık 1mm’dir. Aradaki hata değerleri gerçek arazi şartları dikkate alındığında ihmal edilebilir düzeydedir.

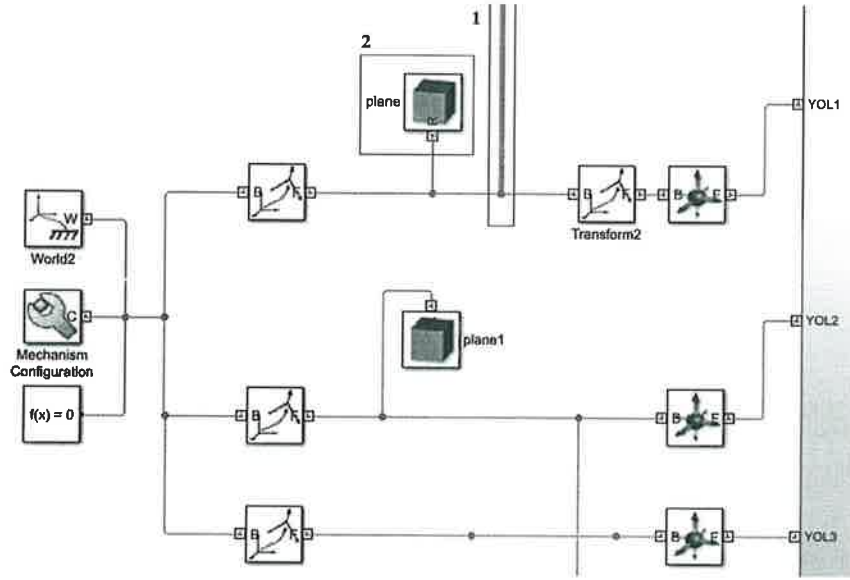


Grafik 5.19.Fuzzy kontrol ile Y ekseninde referans takip hatası

Grafik 5.19’da Fuzzy kontrol ile yapılan Y eksenindeki hata sinyali verilmiştir. Sinyalde başlangıçta sinyali yakalamak için bir salınım olduğu ve  $t=85$ sn değerine kadar hata aralığının az olduğu görülmektedir. Bu noktadan sonra hata değerinin arttığı fakat sinyallerin seyrekleştiği gözlemlenmiştir. Bu değişimin sebebi robotun bacaklarını daha az sayıda ama daha açılı şekilde açmış olmasıdır. Hata aralığı tüm hareket zamanı dikkate alınarak incelendiğinde 0.2cm ile -0.2cm aralığındadır. Bu değerler ihmal edilebilecek kadar küçük değerler olduğu için ihmal edilebilir. Takibin başarılı olduğu gözlemlenmiştir.

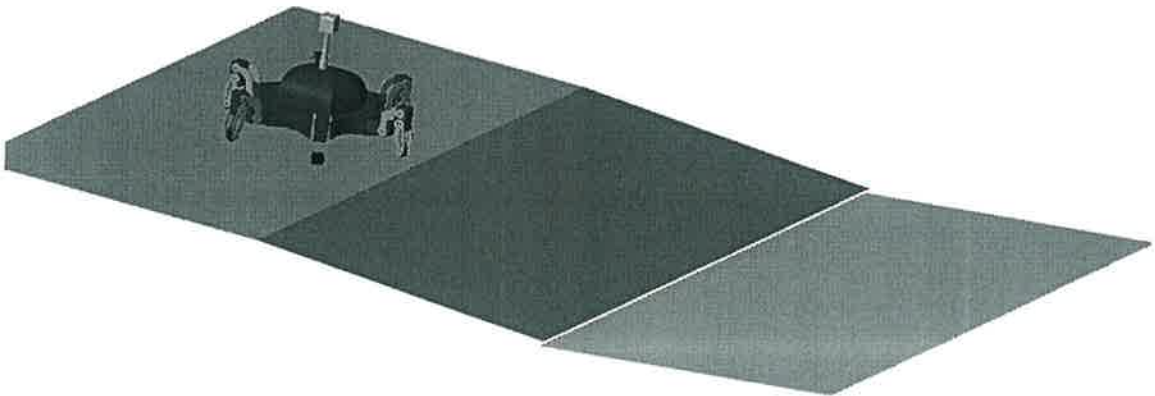
### 5.2.17. Eğimli yol oluşturma

Şuana kadar yapılan analizler düz, sürtünmeli düzlemde gerçekleştirilmiştir. Robotun hareket ettiği yüzeyin sürtünmeli olmasının yanı sıra eğimli olabilme ihtimalide göz önünde bulundurulmalıdır. Bu ihtimal üzerine de analizler yapabilmek için Matlab ortamında birden fazla yol oluşturulmuş bu yolların uzay boşluğunda konumları uç uca eklenecek şekilde ayarlanarak eğimli yollar elde edilmiştir. İşlem Şekil 5.86'da gösterilmiştir.

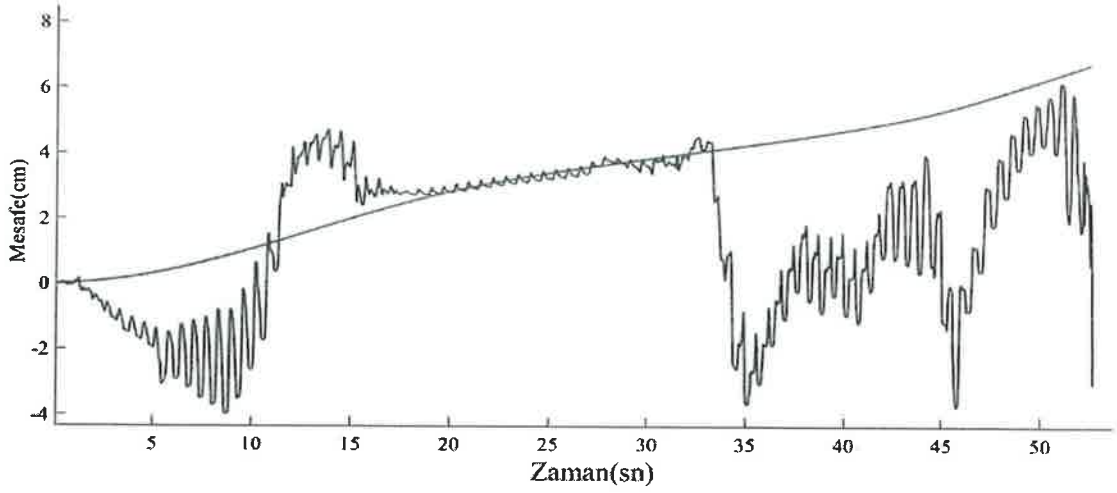


Şekil 5.86. Matlab ortamında eğimli yol oluşturma

Yol oluşturma işlemi yapılırken mekanik parametlerden gövdeye giden bağlantılar paralel yapılarak her bir bağlantıya bir yol atanır ve serbestlik derecesi eklenir. Bu atana yolun uzay boşluğundaki konumu belirlenir. Yolun boşluktaki düzlem derecesi ayarlanarak yolun eğimi ayarlanır. Daha sonra yollar uç uca eklenerek eğimli yol elde edilmiş olur. İşlemlerin sonucunda oluşan eğimli yol Şekil 5.87'de gösterilmiştir.



Şekil 5.87. Matlab ortamında eğimli yol oluşturma

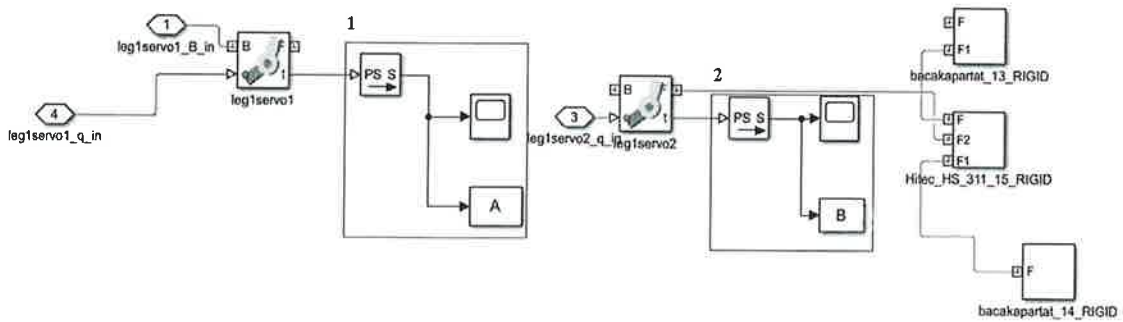


Grafik 5.20. Eğimli yolda elde edilen takip grafiği

Şekil 5.87’de verilen eğimli yolda robotun hareketi sonucunda elde edilen takip sinyali Grafik 5.20’de verilmiştir. Grafikte salınımlar görülmektedir. Bu salınımların sebebi platformlar arası geçişte bazı bacaların boşa çıkmasıdır. Arazi şartlarında bu tür durumlar robotlar için olası durumlardır. Önemli olan robotun sinyali takip etmeyi bırakmamasıdır. Zamanda 10-35 saniye aralığında robotun salınım bittiğinde takibe devam ettiği net bir şekilde gözlemlenmiştir. Takip başarılı şekilde tamamlanmıştır.

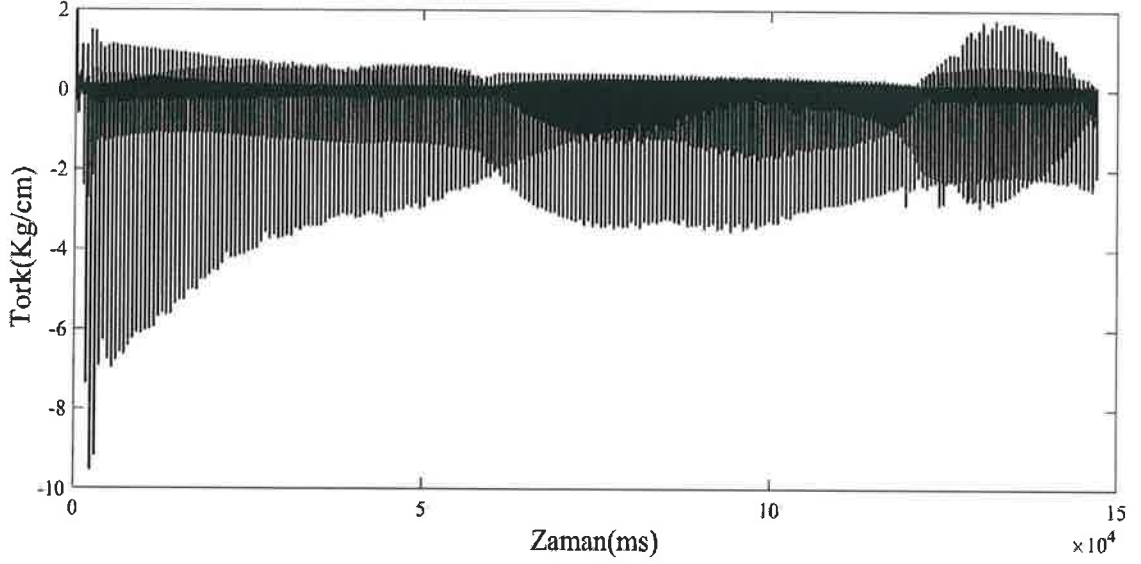
### 5.2.18. Servo motorların tork grafikleri

Bu aşamaya kadar yapılan tüm analizler ve elde edilen tüm grafikler robotun gerçek ağırlığı ve gerçek zemin sürtünme katsayılarına uygun olarak yapılmıştır. Buna bağlı olarak robotun bacalarında bulunan servo motorların ihtiyaç duyduğu tork değerleri de gerçek değerlerdir. Matlabsimulinkteservo motorların tork grafiklerini almakta mümkündür. Şekil 5.88’de gösterilen 1 ve 2 numaralı kutucuklarda servo motorların tork çıkış değerlerinin alınması için kurulan düzenek gösterilmiştir.



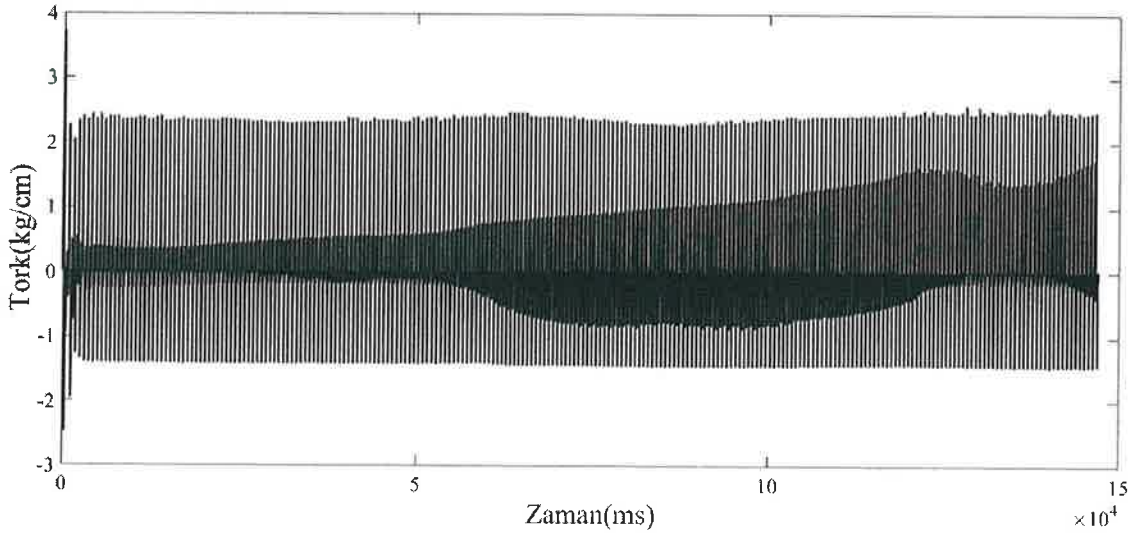
Şekil 5.88. Servo motorların tork değerlerini okuma

1 numaralı kutu ile gösterilen düzener ile iki numaralı kutuda gösterilen düzenerler aynıdır. Bu motorlar robotun bir numaralı ayağında bulunan iki servo motordur. Analizde her iki motorunda tork değerlerinin alınmasının sebebi, motorların zorlanma derecelerinin farklı olmasının düşünülmesidir.



Grafik 5.21. Servo-1 tork grafiği

Grafik 5.21'de gösterilen tork grafiği robotun 1 numaralı bacağında bulunan Servo-1 motoruna aittir. Grafiğin sürekli negatif değerlerde yoğunlaşmasının sebebi robotun hareket esnasında ayağının ileriye havada bir şekilde gidip geriye gelirken yere basarak geri dönmesidir. Robotun harekete başlangıç esnasında -8 ve -9 aralığına kadar bir artış izlenmiştir. Robotun ortalama olarak 3kg/cm tork değerinde çalıştığı gözlemlenmiştir.



Grafik 5.22. Servo-2 tork grafiği



Robotun birinci ayağının ikinci servo motoru olan Servo-2'nin tork grafiği Grafik 5.22'de verilmiştir. Grafikte düzgün bir dağılım izlenmiştir. Bunun nedeni bacağın hareket sırasında yer ile temasının kesilmesi işleminin Servo-2 ile yapılmasıdır. Bu hareket sırasında motora sadece bacağın ağırlığı etki etmektedir. Bu sebeple grafikte tork dağılımının eşit olduğu görülmüştür. Robotun harekete başlangıcı esnasında -2 ve 3.5 aralığında bir yüklenme izlenmiştir. Eylemsizliği yenerek harekete başladıktan sonra ortalama tork 2kg/cm olarak belirlenmiştir.

## 6. SONUÇ

Üzerinde çalışılan tez de altı bacaklı örümceksi bir robotun CAD tasarımı Solidworks programında yapılmış, Matlab fonksiyonları kullanılarak CAD tasarımları Simulink ortamına aktarılmıştır. Robot kontrol benzetim çalışmaları Simulink ortamında gerçekleştirilmiştir. Robotun belli bir yörünge üzerinde otonom olarak hareket etmesi amaçlanmıştır. Robot hareket yörüngesi eğri uydurma yöntemleri ile elde edilmiştir. Robot hareket kontrolleri için PID ve Fuzzy denetleyicileri kullanılmıştır. PID ve Fuzzy denetleyicilerin robot hareket performansları karşılaştırılmıştır. Çalışmadakontroller sırasında iki kontrolör tipi içinde x ekseninde doğrusal referans sinyali ve y ekseninde yörünge oluşturma ile oluşturulan yörünge kullanılmıştır. Robotun hareketi esnasındaki takip grafikleri ve eklemlerde bulunan motorların tork grafikleri elde edilmiştir. Analiz çalışmalarında robotun taşıyacağı yük ile birlikte toplam ağırlığı 4kg olarak dikkate alınmıştır.

Analiz çalışmaları sonucunda tasarım ve analizi yapılan bu robotun 4kg'lık bir kütle ile tanımlanan bir yörüngeyi yaklaşık ortalama 1mm'lik kabul edilebilir bir hata değeri ile takip ettiği gözlemlenmiştir. Fuzzy ve PID performansları incelendiğinde yaklaşık aynı hata oranları ile yörünge izlendiği sonuçlarına varılmıştır. Yörünge takibi sırasında robotun referans yörünge eğrisinin dönüm noktalarında Fuzzy denetleyicinin PID denetleyiciye göre daha iyi bir performans sergilediği görülmüştür.

## 7. ÖNERİLER

Tez çalışmasında ele alınan eklemlili robot Solidworks programında tasarlanmış, hareket kontrol yöntemleri Simulink ortamında denenmiştir. Tasarım çıktıları gerçek bir fiziksel donanıma çevrilip hareket kontrol algoritmalarının bir mikro denetleyici sistem üzerinde düzenlenerek tasarımı yapılan robotun fiziksel bir prototipinin çıkarılması ve gerçek ortam şartlarında hareket kontrol yöntemlerinin denenmesi faydalı olacaktır. Bununla birlikte çalışmada Simulink ortamında kullanılmış olan PID ve Fuzzy kontrol yöntemleri adaptif bir yapı içermemektedir. Bu sebeple PID ve Fuzzy kontrol yöntemleri literatürdeki adaptif algoritmalar ile desteklenerek robotun gerçek fiziksel ortamlardaki dayanıklılığı artırılmalıdır. Yine literatürdeki kayan kipli kontrol, kesir dereceli PID denetleyici yöntemlerinin robot üzerinde denenmesi gerekmektedir.

Tasarımı gerçekleştirilen altı eklemlili robot gerçek hayatta gözlem ve veri toplama uygulamalarında kullanılabileceği düşünülmektedir. Robotun gerçek prototipinde gözlem için kamera, veri toplama için ise uygulamaya özgü algılayıcılarla donatılması gerekmektedir. Bu sayede kamera görüntüsü işlenerek ve makine öğrenmesi algoritmaları kullanılarak engel tanıma, bitki yapraklarının analizi, mayın tespiti ve görüntü aktarma gibi işlemlerde kullanılabilir.

## 8. KAYNAKLAR

- Bapat A., 2016. Design, Prototyping and Testing Of An Autonomous Hexapod Robot With C Shaped Compliant Legs, The University Of Texas At San Antonio Department Of Mechanical Engineering, U.S.A.
- Bingül Z., Küçük S., 2009. Robot Kinematığı, Birsen Yayınevi, İstanbul.
- Boston Dynamics, 2016. <https://www.bostondynamics.com/atlas> (11.5.2017)
- Böttcher S., 2004. Human Robot Interaction, pp 12-45, Almanya.
- Buehler M., Saranlı U., Koditschek D.E., 2001. RHex: A Simple and Highly Mobile Hexapod Robot, The International Journal of Robotics Research, Korea
- Bulanık mantık nedir, 2007. <https://www.kontrolkalemi.com/forum/konu/bulanik-mantik-fuzzy-logic-nedir.3068/> (14.04.2018)
- Caponetto R., Fortuna L., Porto D., 2002, Parameter Tuning of A Non-Integer Order PID Controller, 15th International Symposium on Mathematical Theory of Networks and Systems, Notre Dame, Indiana.
- Common Walking Gaits For Hexapod, <https://hexyrobot.wordpress.com/2015.11.20/common-walking-gaits-for-hexapods/>, (21.07.2017)
- Dormand, J. R.; Prince, P. J., 1980. "A family of embedded Runge-Kutta formulae", Journal of Computational and Applied Mathematics, pp. 19–26 Cleveland UK.
- Earnest L., 1980. Stanford Artificial Intelligence Laboratory, Stanford University Retrieved U.S.A
- Ferdinand P., 2002. Vector Mechanics For Engineers: Statics, Late Of Lehigh University, U.S.A.
- Hardarson F., 1998. Locomotion For Difficult Terrain, Royal Inst Technol Tech. Rep., Stockholm, Sweden.
- Solidworks yardım menüsü <http://help.solidworks.com/2010/English/SolidWorks/sldworks/LegacyHelp/Sldworks/Overview/StartPage.htm> (8.11.2017)
- Isaac A., 1950. Robotik El Kitabı 56. Baskı., U.S.A.
- Mamidi S. V.S., 2012. Manufacturing An Involute Spline Cutting Tool On Wire Edm Using Solidworks And Esprit, California State University, U.S.A.
- Mahajan A., Figueroa F., 1997. Four Legged Intelligent Mobile Autonomous Robot, Robotics and Computer-Integrated Manufacturing 13, sayfa 51–61, Lake Superior State University, U.S.A.
- Matlab & Simulink Mathworks, Choose A Solver, <https://www.mathworks.com/help/simulink/ug/choose-a-solver.html>. (02.27.2018)
- MATLAB and Simulink Robotics Arena: Walking Robot, <https://www.mathworks.com/matlabcentral/fileexchange/64227-matlab-and-simulink-robotics-arena-walking-robot> (20.8.2018)
- Mobil robotların tarihçesi, 2008. [https://en.wikipedia.org/wiki/Mobile\\_robot](https://en.wikipedia.org/wiki/Mobile_robot), (26.12.2018).
- Moncalı K. K., Aydın Ö. C., 2016. Örümcek Robot Sistem Tasarımı ve Gerçeklenmesi, Kocaeli Üniversitesi, Türkiye
- Özel H., 2018. <https://medium.com/matlab-nedir> (20.03.2019)
- Küçüköncü T., 2007. Çanakkale On Sekiz Mart Üniversitesi, Bilgisayar Mühendisliği Bölümü, Bulanık mantık, Türkiye.

- Portal Netcad, 2013. <http://portal.netcad.com.tr/pages/viewpage.action?pageId=104794233>. (24.11.2018)
- Robot platform, [http://www.robotplatform.com/knowledge/Classification\\_of\\_Robots/legged\\_robots.html](http://www.robotplatform.com/knowledge/Classification_of_Robots/legged_robots.html). (10.10.2018)
- Robot Tarihi, 2001. <http://www.moment-expo.com/m-oden-gunumuze-robot-tarihi>, (19.11.2018).
- Shahriari M., 2013. Design, Implementation and Control Of A Hexapod Robot Using Reinforcement Learning Approach, Presented to Sharif University of Technology, International Campus, Kish Island
- Shampine, 1986. Hairer, Nørsett, Wanner 2008, s. 178–179 "Some Practical Runge-Kutta Formulas", American Mathematical Society, 46 (173): 135–150, U.S.A.
- Sürtünme Kuvveti Faktörleri, <https://www.fizikbilimi.gen.tr/surtunme-kuvveti>. (09.09.2017)
- Villumsen S., 2010, Modelleri göç kontrol seksbenet mobil robot, USA.
- Yamanol İ., 2012. Robotların Tarihi, Bilim Kurgu Kulübü, Türkiye
- Zamri R., S.Kuswadi, Prabuwono A.s., Besari A.R.A., 2009. The Study on Optimal Gait for Five-Legged Robot with Reinforcement Learning, from book Intelligent Robotics and Applications, Second International Conference, Malaysia.
- Wikipedia, Dormand–Prince method [https://en.wikipedia.org/wiki/Dormand–Prince\\_method](https://en.wikipedia.org/wiki/Dormand–Prince_method) (11.11.2018)
- Woollaston, Victoria, 2013. "Plant bot: The world's first robot that can turn your household plants into light-seeking 'triffid' drones", New Jersey University, U.S.A.

## 9. EKLER

EK-1 EvalSmooth trajectoryoluřturma kodları

```
function out = evalSmoothTrajectory(params,t)
tEff = mod(t,params.gaitPeriod);
indices = find(tEff >= params.gaitTime(1:end-1));
idx = indices(end);
dt = tEff - params.gaitTime(idx);
out = zeros(1,1);
out(1) = params.a0_ merkez(idx) + params.a1_ merkez(idx)*dt + ...
params.a2_ merkez(idx)*dt^2 + params.a3_ merkez(idx)*dt^3;
end (Anonim).
```

```
function out = smoothTraj(t,curveData)
out = evalSmoothTrajectory(curveData,t);
end (Anonim).
```

EK-2 Parça ve montaj veri dosyaları

```
smiData = struct;
smiData.RigidTransform = struct;
smiData.RigidTransform(1).translation = [-4.79658495188 -0.476427209 -0.02918];
smiData.RigidTransform(1).angle = 3.1415926535897931;
smiData.RigidTransform(1).axis = [1 0 0];
smiData.RigidTransform(1).ID = 'B[Hitec HS-311-1:-:HSaparat-18]';
smiData.RigidTransform(2).translation = [15.772239759134301 -22.64227
smiData.RigidTransform(2).angle = 3.1415926535897931;
smiData.RigidTransform(2).axis = [1 -9.2444637330586336E-33
smiData.RigidTransform(2).ID = 'F[Hitec HS-311-1:-:HSaparat-18]';
smiData.RigidTransform(3).translation = [0 0 2.0004];
smiData.RigidTransform(3).angle = 0;
smiData.RigidTransform(3).axis = [0 0 0];
smiData.RigidTransform(3).ID = 'B[Hitec HS-311-1:-:HSaparat-19]';
smiData.RigidTransform(4).translation = [20.568824360043614 -22.16363 0];
smiData.RigidTransform(4).angle = 2.3069475408519375E-14;
smiData.RigidTransform(4).axis = [-0.052937715551029649 -0.9985978160762411
smiData.RigidTransform(4).ID = 'F[Hitec HS-311-1:-:HSaparat-19]';
smiData.RigidTransform(5).translation = [2.1641296848049469 -8.183967 0];
smiData.RigidTransform(5).angle = 0;
smiData.RigidTransform(5).axis = [0 0 0];
smiData.RigidTransform(5).ID = 'B[ustcerceve-1:-:Hitec HS-311-1]';
smiData.RigidTransform(6).translation = [-7.1465846009095184 -1.571043034
```

```

smiData.RigidTransform(6).angle = 2.0943951023931948;
smiData.RigidTransform(6).axis = [-0.57735026918962562 -0.5773502691896254 ...
smiData.RigidTransform(6).ID = 'F[ustcerceve-1:-:HITEC HS-311-1]';
smiData.RigidTransform(7).translation = [0 0 2.000018];
smiData.RigidTransform(7).angle = 0;
smiData.RigidTransform(7).axis = [0 0 0];
smiData.RigidTransform(7).ID = 'B[HITEC HS-311-2:-:HSAPARAT-14]';
smiData.RigidTransform(8).translation = [20.5688206 -22.12266 -7.1056010019E-15];
smiData.RigidTransform(8).angle = 2.5207197528726529E-15;
smiData.RigidTransform(8).axis = [0.13213166874579316 -0.022021944790965527 ...
smiData.RigidTransform(8).ID = 'F[HITEC HS-311-2:-:HSAPARAT-14]';
smiData.RigidTransform(9).translation = [-5.2965179 -0.476426931 -0.0299997824];
smiData.RigidTransform(9).angle = 3.1415926535897931;
smiData.RigidTransform(9).axis = [1 0 0];
smiData.RigidTransform(9).ID = 'B[HITEC HS-311-2:-:HSAPARAT-15]';
smiData.RigidTransform(10).translation = [15.27220066 -22.6400202 0.170000171];
smiData.RigidTransform(10).angle = 3.1415926535897931;
smiData.RigidTransform(10).axis = [1 7.7521322694451555E-14
smiData.RigidTransform(10).ID = 'F[HITEC HS-311-2:-:HSAPARAT-15]';
smiData.RigidTransform(11).translation = [2.1641047449 8.18399326 0];
smiData.RigidTransform(11).angle = 0;
smiData.RigidTransform(11).axis = [0 0 0];
smiData.RigidTransform(11).ID = 'B[ustcerceve-1:-:HITEC HS-311-2]';
smiData.RigidTransform(12).translation = [-7.1465273 -1.5713087 1.000000004];
smiData.RigidTransform(12).angle = 2.0943951023931953;
smiData.RigidTransform(12).axis = [-0.57735026918962584 -0.57735026918962529 ...
smiData.RigidTransform(12).ID = 'F[ustcerceve-1:-:HITEC HS-311-2]';
smiData.RigidTransform(13).translation = [-4.7969095188 -0.472857209 -
smiData.RigidTransform(13).angle = 3.1415926535897931;
smiData.RigidTransform(13).axis = [1 0 0];
smiData.RigidTransform(13).ID = 'B[HITEC HS-311-3:-:HSAPARAT-16]';
smiData.RigidTransform(14).translation = [15.772083 -22.6400673
smiData.RigidTransform(14).angle = 3.1415926535897927;
smiData.RigidTransform(14).axis = [-1 -1.2241076208230336E-14
smiData.RigidTransform(14).ID = 'F[HITEC HS-311-3:-:HSAPARAT-16]';
smiData.RigidTransform(15).translation = [-5.296595179 -0.472857209 2.03000007];
smiData.RigidTransform(15).angle = 0;
smiData.RigidTransform(15).axis = [0 0 0];
smiData.RigidTransform(15).ID = 'B[HITEC HS-311-3:-:HSAPARAT-17]';
smiData.RigidTransform(16).translation = ...[15. -0.120000000001];
smiData.RigidTransform(16).angle = 6.6263342635487934E-14;
smiData.RigidTransform(16).axis = [0.0033509418042263575

```

smiData.RigidTransform(16).ID = 'F[Hitec HS-311-3:-:HSaparat-17]';  
smiData.RigidTransform(17).translation = [-2.56098 1.80411258794E-14 0];  
smiData.RigidTransform(17).angle = 0;  
smiData.RigidTransform(17).axis = [0 0 0];  
smiData.RigidTransform(17).ID = 'B[ustcerceve-1:-:Hitec HS-311-3]';  
smiData.RigidTransform(18).translation = [-7.14690952 -1.571043003 1.000000027];  
smiData.RigidTransform(18).angle = 2.0943951023931957;  
smiData.RigidTransform(18).axis = [-0.57735026918962584 -0.57735026918962573 ...  
smiData.RigidTransform(18).ID = 'F[ustcerceve-1:-:Hitec HS-311-3]';  
smiData.RigidTransform(19).translation = ...  
smiData.RigidTransform(19).angle = 3.1415926535897931;  
smiData.RigidTransform(19).axis = [1 0 0];  
smiData.RigidTransform(19).ID = 'B[Hitec HS-311-4:-:HSaparat-20]';  
smiData.RigidTransform(20).translation = [20.56880647 -22.1636525  
smiData.RigidTransform(20).angle = 3.1415926535897931;  
smiData.RigidTransform(20).axis = [1 5.0182080713057076E-14 6.38378239159465E  
smiData.RigidTransform(20).ID = 'F[Hitec HS-311-4:-:HSaparat-20]';  
smiData.RigidTransform(21).translation = [-5.2965179 -0.4767209 2.030000007];  
smiData.RigidTransform(21).angle = 0;  
smiData.RigidTransform(21).axis = [0 0 0];  
smiData.RigidTransform(21).ID = 'B[Hitec HS-311-4:-:HSaparat-21]';  
smiData.RigidTransform(22).translation = [15. --0.11999999389];  
smiData.RigidTransform(22).angle = 2.4487906147411006E-15;  
smiData.RigidTransform(22).axis = [0.068006408016791711 -0.022668802672263903  
smiData.RigidTransform(22).ID = 'F[Hitec HS-311-4:-:HSaparat-21]';  
smiData.RigidTransform(23).translation = ...[11. 0];  
smiData.RigidTransform(23).angle = 0;  
smiData.RigidTransform(23).axis = [0 0 0];  
smiData.RigidTransform(23).ID = 'B[ustcerceve-1:-:Hitec HS-311-4]';  
smiData.RigidTransform(24).translation = ...[-7.1.];  
smiData.RigidTransform(24).angle = 2.0943951023931953;  
smiData.RigidTransform(24).axis = [-0.57735026918962584 -0.57735026918962584 ...  
smiData.RigidTransform(24).ID = 'F[ustcerceve-1:-:Hitec HS-311-4]';  
smiData.RigidTransform(25).translation = ...  
smiData.RigidTransform(25).angle = 0;  
smiData.RigidTransform(25).axis = [0 0 0];  
smiData.RigidTransform(25).ID = 'B[Hitec HS-311-5:-:HSaparat-13]';  
smiData.RigidTransform(26).angle = 1.9918567808888279E-15;  
smiData.RigidTransform(26).ID = 'F[Hitec HS-311-5:-:HSaparat-13]';  
smiData.RigidTransform(27).angle = 0;  
smiData.RigidTransform(27).axis = [0 0 0];  
smiData.RigidTransform(27).ID = 'B[ustcerceve-1:-:Hitec HS-311-5]';



```

smiData.RigidTransform(28).angle = 2.0943951023931957;
smiData.RigidTransform(28).axis = [-0.57735026918962584 -0.57735026918962573 ...
smiData.RigidTransform(28).ID = 'F[ustcerceve-1:-:Hitec HS-311-5]';
smiData.RigidTransform(29).angle = 3.1415926535897931;
smiData.RigidTransform(29).axis = [1 0 0];
smiData.RigidTransform(29).ID = 'B[Hitec HS-311-5:-:HSaparat-4]';
smiData.RigidTransform(30).angle = 3.1415926535897931;
smiData.RigidTransform(30).ID = 'F[Hitec HS-311-5:-:HSaparat-4]';
smiData.RigidTransform(31).angle = 0;
smiData.RigidTransform(31).axis = [0 0 0];
smiData.RigidTransform(31).ID = 'B[ustcerceve-1:-:Hitec HS-311-6]';
smiData.RigidTransform(32).angle = 2.0943951023931957;
smiData.RigidTransform(32).axis = [-0.57735026918962584 -0.57735026918962573 ...
smiData.RigidTransform(32).ID = 'F[ustcerceve-1:-:Hitec HS-311-6]';
smiData.RigidTransform(33).angle = 0;
smiData.RigidTransform(33).axis = [0 0 0];
smiData.RigidTransform(33).ID = 'B[Hitec HS-311-6:-:HSaparat-1]';
smiData.RigidTransform(34).angle = 1.4455449122366035E-13;
smiData.RigidTransform(34).ID = 'F[Hitec HS-311-6:-:HSaparat-1]';
smiData.RigidTransform(35).angle = 3.1415926535897931;
smiData.RigidTransform(35).axis = [1 0 0];
smiData.RigidTransform(35).ID = 'B[Hitec HS-311-6:-:HSaparat-2]';
smiData.RigidTransform(36).angle = 3.1415926535897931;
smiData.RigidTransform(36).ID = 'F[Hitec HS-311-6:-:HSaparat-2]';
smiData.RigidTransform(37).angle = 2.0943951023931953;
smiData.RigidTransform(37).axis = [0.57735026918962584 -0.57735026918962584 ...
smiData.RigidTransform(37).ID = 'B[Hitec HS-311-7:-:bacakapartat-1]';
smiData.RigidTransform(38).angle = 3.1415926535897927;
smiData.RigidTransform(38).ID = 'F[Hitec HS-311-7:-:bacakapartat-1]';
smiData.RigidTransform(39).angle = 2.0943951023931953;
smiData.RigidTransform(39).ID = 'B[Hitec HS-311-7:-:bacakapartat-2]';
smiData.RigidTransform(40).angle = 1.2967861472664216E-13;
smiData.RigidTransform(40).ID = 'F[Hitec HS-311-7:-:bacakapartat-2]';
smiData.RigidTransform(41).angle = 2.0943951023931953;
smiData.RigidTransform(41).axis = [0.57735026918962584 -0.57735026918962584 ...
smiData.RigidTransform(41).ID = 'B[Hitec HS-311-7:-:HSaparat-1]';
smiData.RigidTransform(42).angle = 1.1443916996305594E-16;
smiData.RigidTransform(42).axis = [-0.38036386164108238 0.92483692225034109 ...
smiData.RigidTransform(42).ID = 'F[Hitec HS-311-7:-:HSaparat-1]';
smiData.RigidTransform(43).angle = 2.0943951023931953;
smiData.RigidTransform(43).axis = [-0.57735026918962584 -0.57735026918962584 ...
smiData.RigidTransform(43).ID = 'B[Hitec HS-311-7:-:HSaparat-2]';

```

```

smiData.RigidTransform(44).angle = 3.1415926535897931;
smiData.RigidTransform(44).ID = 'F[Hitec HS-311-7:-:HSaparat-2]';
smiData.RigidTransform(45).angle = 2.0943951023931953;
smiData.RigidTransform(45).axis = [-0.57735026918962584 -0.57735026918962584 ...
smiData.RigidTransform(45).ID = 'B[Hitec HS-311-1:-:ust cerceve-1]';
smiData.RigidTransform(46).angle = 1.5700924586837751E-15;
smiData.RigidTransform(46).ID = 'F[Hitec HS-311-1:-:ust cerceve-1]';
smiData.RigidTransform(47).angle = 2.0943951023931953;
smiData.RigidTransform(47).axis = [-0.57735026918962584 -0.57735026918962584 ...
smiData.RigidTransform(47).ID = 'B[Hitec HS-311-6:-:ust cerceve-1]';
smiData.RigidTransform(48).angle = 0;
smiData.RigidTransform(48).axis = [0 0 0];
smiData.RigidTransform(48).ID = 'F[Hitec HS-311-6:-:ust cerceve-1]';
smiData.RigidTransform(49).angle = 2.0943951023931953;
smiData.RigidTransform(49).axis = [-0.57735026918962584 -0.57735026918962584 ...
smiData.RigidTransform(49).ID = 'B[antenkaybetme-1:-:camera tutck-2]';
smiData.RigidTransform(50).angle = 3.1415926535897931;
smiData.RigidTransform(50).axis = [-0.6553101801596255 -0.75535989288494731 ...
smiData.RigidTransform(50).ID = 'F[antenkaybetme-1:-:camera tutck-2]';
smiData.RigidTransform(51).angle = 2.0943951023931957;
smiData.RigidTransform(51).axis = [0.57735026918962584 -0.57735026918962573 ...
smiData.RigidTransform(51).ID = 'B[camera-2:-:camera tutck-2]';
smiData.RigidTransform(52).angle = 6.19491587045462E-16;
smiData.RigidTransform(52).axis = [0.94463176228977885 -0.09648419790895886 ...
smiData.RigidTransform(52).ID = 'F[camera-2:-:camera tutck-2]';
smiData.RigidTransform(53).translation = [0 0 0.149999999999999875];
smiData.RigidTransform(53).angle = 1.1823678321455939E-16;
smiData.RigidTransform(53).axis = [-0 -1 0];
smiData.RigidTransform(53).ID = 'B[bacakapartat-13:-:pabuc-3]';
smiData.RigidTransform(54).translation = ...
smiData.RigidTransform(54).angle = 3.1415926535897931;
smiData.RigidTransform(54).axis = [-0.63152544803602739 0.77535514990415511 ...
smiData.RigidTransform(54).ID = 'F[bacakapartat-13:-:pabuc-3]';
smiData.RigidTransform(55).translation = [0 0 0.14999999999999997];
smiData.RigidTransform(55).angle = 0;
smiData.RigidTransform(55).axis = [0 0 0];
smiData.RigidTransform(55).ID = 'B[bacakapartat-15:-:pabuc-2]';
smiData.RigidTransform(56).angle = 3.1415926535897931;
smiData.RigidTransform(56).axis = [0.63050855657310645 -0.77618229823160612 ...
smiData.RigidTransform(56).ID = 'F[bacakapartat-15:-:pabuc-2]';
smiData.RigidTransform(57).angle = 0;
smiData.RigidTransform(57).axis = [0 0 0];

```

```

smiData.RigidTransform(57).ID = 'B[bacakapartat-17:-:pabuc-5]';
smiData.RigidTransform(58).angle = 3.1415926535897931;
smiData.RigidTransform(58).axis = [0.631525448036028 -0.77535514990415477 ...
smiData.RigidTransform(58).ID = 'F[bacakapartat-17:-:pabuc-5]';
smiData.RigidTransform(59).angle = 0;
smiData.RigidTransform(59).axis = [0 0 0];
smiData.RigidTransform(59).ID = 'B[bacakapartat-19:-:pabuc-6]';
smiData.RigidTransform(60).angle = 3.1415926535897922;
smiData.RigidTransform(60).axis = [-0.63152544803602806 0.77535514990415477 ...
smiData.RigidTransform(60).ID = 'F[bacakapartat-19:-:pabuc-6]';
smiData.RigidTransform(61).angle = 2.0943951023931953;
smiData.RigidTransform(61).axis = [0.57735026918962584 -0.57735026918962584 ...
smiData.RigidTransform(61).ID = 'B[antenkaybetme-1:-:guc kutusu-3]';
smiData.RigidTransform(62).angle = 3.1415926535897931;
smiData.RigidTransform(62).axis = [0.70710678118655068 0.70710678118654435 ...
smiData.RigidTransform(62).ID = 'F[antenkaybetme-1:-:guc kutusu-3]';
smiData.RigidTransform(63).angle = 2.0943951023931953;
smiData.RigidTransform(63).axis = [-0.57735026918962584 -0.57735026918962584 ...
smiData.RigidTransform(63).ID = 'B[ustcerceve-1:-:guc kutusu-3]';
smiData.RigidTransform(64).angle = 2.0943951023931953;
smiData.RigidTransform(64).axis = [-0.57735026918962584 -0.57735026918962584 ...
smiData.RigidTransform(64).ID = 'F[ustcerceve-1:-:guc kutusu-3]';
smiData.RigidTransform(65).angle = 0;
smiData.RigidTransform(65).axis = [0 0 0];
smiData.RigidTransform(65).ID = 'B[bacakapartat-2:-:pabuc-7]';
smiData.RigidTransform(66).angle = 1.7745545386769159;
smiData.RigidTransform(66).ID = 'F[bacakapartat-2:-:pabuc-7]';
smiData.RigidTransform(67).angle = 0;
smiData.RigidTransform(67).axis = [0 0 0];
smiData.RigidTransform(67).ID = 'B[bacakapartat-4:-:pabuc-4]';
smiData.RigidTransform(68).angle = 3.1415926535897927;
smiData.RigidTransform(68).axis = [-0.77535514990415477 -0.631525448036028 ...
smiData.RigidTransform(68).ID = 'F[bacakapartat-4:-:pabuc-4]';
smiData.RigidTransform(69).angle = 0;
smiData.RigidTransform(69).axis = [0 0 0];
smiData.RigidTransform(69).ID = 'B[HSaparat-13:-:Hitec HS-311-12]';
smiData.RigidTransform(70).angle = 2.0943951023931957;
smiData.RigidTransform(70).axis = [0.577350269189626 -0.57735026918962573 ...
smiData.RigidTransform(70).ID = 'F[HSaparat-13:-:Hitec HS-311-12]';
smiData.RigidTransform(71).translation = ...
smiData.RigidTransform(71).angle = 3.1415926535897931;
smiData.RigidTransform(71).axis = [1 0 0];

```

smiData.RigidTransform(71).ID = 'B[bacakapartat-4::-Hitec HS-311-12]';  
smiData.RigidTransform(72).angle = 2.0943951023931882;  
smiData.RigidTransform(72).axis = [0.57735026918963062 -0.5773502691896234 ...  
smiData.RigidTransform(72).ID = 'F[bacakapartat-4::-Hitec HS-311-12]';  
smiData.RigidTransform(73).angle = 0;  
smiData.RigidTransform(73).axis = [0 0 0];  
smiData.RigidTransform(73).ID = 'B[bacakapartat-5::-Hitec HS-311-12]';  
smiData.RigidTransform(74).angle = 2.0943951023931882;  
smiData.RigidTransform(74).axis = [-0.57735026918963051 -0.57735026918962329 ...  
smiData.RigidTransform(74).ID = 'F[bacakapartat-5::-Hitec HS-311-12]';  
smiData.RigidTransform(75).translation = ...  
smiData.RigidTransform(75).angle = 3.1415926535897931;  
smiData.RigidTransform(75).axis = [1 0 5.91183916072797E-17];  
smiData.RigidTransform(75).ID = 'B[bacakapartat-13::-Hitec HS-311-15]';  
smiData.RigidTransform(76).angle = 2.094395102393209;  
smiData.RigidTransform(76).axis = [0.57735026918961507 -0.57735026918963073 ...  
smiData.RigidTransform(76).ID = 'F[bacakapartat-13::-Hitec HS-311-15]';  
smiData.RigidTransform(77).angle = 1.1357328464819463E-16;  
smiData.RigidTransform(77).axis = [-0 -1 0];  
smiData.RigidTransform(77).ID = 'B[bacakapartat-14::-Hitec HS-311-15]';  
smiData.RigidTransform(78).angle = 2.0943951023931606;  
smiData.RigidTransform(78).axis = [-0.57735026918964871 -0.57735026918961407 ...  
smiData.RigidTransform(78).ID = 'F[bacakapartat-14::-Hitec HS-311-15]';  
smiData.RigidTransform(79).angle = 0;  
smiData.RigidTransform(79).axis = [0 0 0];  
smiData.RigidTransform(79).ID = 'B[HSaparat-14::-Hitec HS-311-15]';  
smiData.RigidTransform(80).angle = 2.0943951023931957;  
smiData.RigidTransform(80).axis = [0.57735026918962584 -0.57735026918962584 ...  
smiData.RigidTransform(80).ID = 'F[HSaparat-14::-Hitec HS-311-15]';  
smiData.RigidTransform(81).angle = 0;  
smiData.RigidTransform(81).axis = [0 0 0];  
smiData.RigidTransform(81).ID = 'B[bacakapartat-15::-Hitec HS-311-16]';  
smiData.RigidTransform(82).angle = 2.0943951023932175;  
smiData.RigidTransform(82).axis = [-0.57735026918961119 -0.57735026918963261 ...  
smiData.RigidTransform(82).ID = 'F[bacakapartat-15::-Hitec HS-311-16]';  
smiData.RigidTransform(83).angle = 0;  
smiData.RigidTransform(83).axis = [0 0 0];  
smiData.RigidTransform(83).ID = 'B[bacakapartat-16::-Hitec HS-311-16]';  
smiData.RigidTransform(84).translation = [-5. 0.75000000000190781];  
smiData.RigidTransform(84).angle = 2.0943951023931464;  
smiData.RigidTransform(84).axis = [-0.57735026918965915 -0.5773502691896093 ...  
smiData.RigidTransform(84).ID = 'F[bacakapartat-16::-Hitec HS-311-16]';

```

smiData.RigidTransform(85).translation = ...[18. 0];
smiData.RigidTransform(85).angle = 0;
smiData.RigidTransform(85).axis = [0 0 0];
smiData.RigidTransform(85).ID = 'B[HSaparat-16:-:Hitec HS-311-16]';
smiData.RigidTransform(86).translation = [-7.00000000000005418];
smiData.RigidTransform(86).angle = 2.0943951023931953;
smiData.RigidTransform(86).axis = [0.57735026918962584 -0.577350269189626 ...
smiData.RigidTransform(86).ID = 'F[HSaparat-16:-:Hitec HS-311-16]';
smiData.RigidTransform(87).translation = ...[-7. 1.000000009];
smiData.RigidTransform(87).angle = 2.0943951023931953;
smiData.RigidTransform(87).axis = [0.57735026918962584 -0.57735026918962584 ...
smiData.RigidTransform(87).ID = 'B[Hitec HS-311-16:-:HSaparat-17]';
smiData.RigidTransform(88).translation = [18. 0];
smiData.RigidTransform(88).angle = 1.0928757898653883E-16;
smiData.RigidTransform(88).axis = [-0.063716937823560443 0.99796801142841685
smiData.RigidTransform(88).ID = 'F[Hitec HS-311-16:-:HSaparat-17]';
smiData.RigidTransform(89).translation = [11.436383680156133 -17.8171];
smiData.RigidTransform(89).angle = 0;
smiData.RigidTransform(89).axis = [0 0 0];
smiData.RigidTransform(89).ID = 'B[bacakapartat-17:-:Hitec HS-311-18]';
smiData.RigidTransform(90).translation = [-5.396580254 1.078856987 0.750038023];
smiData.RigidTransform(90).angle = 2.0943951023931562;
smiData.RigidTransform(90).axis = [-0.57735026918965149 -0.57735026918961208 ...
smiData.RigidTransform(90).ID = 'F[bacakapartat-17:-:Hitec HS-311-18]';
smiData.RigidTransform(91).translation = [11.4363856132 -17.817107659884645 0];
smiData.RigidTransform(91).angle = 0;
smiData.RigidTransform(91).axis = [0 0 0];
smiData.RigidTransform(91).ID = 'B[bacakapartat-18:-:Hitec HS-311-18]';
smiData.RigidTransform(92).translation = [-5.396584600914677 -1.371];
smiData.RigidTransform(92).angle = 2.0943951023933414;
smiData.RigidTransform(92).axis = [-0.57735026918952825 -0.57735026918967447 ...
smiData.RigidTransform(92).ID = 'F[bacakapartat-18:-:Hitec HS-311-18]';
smiData.RigidTransform(93).translation = [18.213093700760449 -18.96167];
smiData.RigidTransform(93).angle = 0;
smiData.RigidTransform(93).axis = [0 0 0];
smiData.RigidTransform(93).ID = 'B[HSaparat-18:-:Hitec HS-311-18]';
smiData.RigidTransform(94).translation = [-7..1288437322];
smiData.RigidTransform(94).angle = 2.0943951023931953;
smiData.RigidTransform(94).axis = [0.57735026918962584 -0.57735026918962551 ...
smiData.RigidTransform(94).ID = 'F[HSaparat-18:-:Hitec HS-311-18]';
smiData.RigidTransform(95).translation = [-7.146584095166 -1.221];
smiData.RigidTransform(95).angle = 2.0943951023931953;

```

```

smiData.RigidTransform(95).axis = [0.57735026918962584 -0.57735026918962584 ...
smiData.RigidTransform(95).ID = 'B[Hitec HS-311-18::-HSaparat-19]';
smiData.RigidTransform(96).translation = [18.236106 -18.9648291 -3.5527005];
smiData.RigidTransform(96).angle = 2.2887833992611192E-16;
smiData.RigidTransform(96).axis = [-0.91996807299210082 -0.39199329672227878 ...
smiData.RigidTransform(96).ID = 'F[Hitec HS-311-18::-HSaparat-19]';
smiData.RigidTransform(97).translation = [11.436383680156137 -17];
smiData.RigidTransform(97).angle = 0;
smiData.RigidTransform(97).axis = [0 0 0];
smiData.RigidTransform(97).ID = 'B[bacakapartat-19::-Hitec HS-311-20]';
smiData.RigidTransform(98).translation = [-5.39659113747];
smiData.RigidTransform(98).angle = 2.09439510239325;
smiData.RigidTransform(98).axis = [-0.57735026918959031 -0.577350269189643 ...
smiData.RigidTransform(98).ID = 'F[bacakapartat-19::-Hitec HS-311-20]';
smiData.RigidTransform(99).translation = [11.43638368015613]
smiData.RigidTransform(99).angle = 0;
smiData.RigidTransform(99).axis = [0 0 0];
smiData.RigidTransform(99).ID = 'B[bacakapartat-20::-Hitec HS-311-20]';
smiData.RigidTransform(100).translation = [-5.75000000000057288];
smiData.RigidTransform(100).angle = 2.0943951023931779;
smiData.RigidTransform(100).axis = [-0.57735026918963828 -0.57735026918961962
smiData.RigidTransform(100).ID = 'F[bacakapartat-20::-Hitec HS-311-20]';
smiData.RigidTransform(101).translation = [18.2160457 -18.791148 0];
smiData.RigidTransform(101).angle = 0;
smiData.RigidTransform(101).axis = [0 0 0];
smiData.RigidTransform(101).ID = 'B[HSaparat-20::-Hitec HS-311-20]';
smiData.RigidTransform(102).translation = [-7.146809 1.128895
smiData.RigidTransform(102).angle = 2.0943951023931948;
smiData.RigidTransform(102).axis = [0.57735026918962562 -0.57735026918962551
smiData.RigidTransform(102).ID = 'F[HSaparat-20::-Hitec HS-311-20]';
smiData.RigidTransform(103).translation = [-7.1465814 -1.2211562677043029 1.000];
smiData.RigidTransform(103).angle = 2.0943951023931953;
smiData.RigidTransform(103).axis = [0.57735026918962584 -0.57735026918962584
smiData.RigidTransform(103).ID = 'B[Hitec HS-311-20::-HSaparat-21]';
smiData.RigidTransform(104).translation = [18.21376 -18.961971 -1.06];
smiData.RigidTransform(104).angle = 4.47545209131181E-16;
smiData.RigidTransform(104).axis = [0.91060031139618336 -0.41328812332944453
smiData.RigidTransform(104).ID = 'F[Hitec HS-311-20::-HSaparat-21]';
smiData.RigidTransform(105).translation = [-6.24.146991116044735];
smiData.RigidTransform(105).angle = 1.8029001865229415;
smiData.RigidTransform(105).axis = [-0.79119061155601067 -0.432445034764897 ...
smiData.RigidTransform(105).ID = ['AssemblyGround[antenkaybetme-1:dista' ...

```

```

smiData.RigidTransform(106).translation = [-4.7526623022299876]
smiData.RigidTransform(106).axis = [-0.91683985733567075 -0.28231602505056891
smiData.RigidTransform(106).ID = ['AssemblyGround[antenkaybetme-1:dista' ...
smiData.RigidTransform(107).translation = [-6. 18.385104684510058];
smiData.RigidTransform(107).angle = 2.1779943356443439;
smiData.RigidTransform(107).axis = [-0.52289978582121 0.60273369492178408 ...
smiData.RigidTransform(107).ID = ['AssemblyGround[antenkaybetme-1:dista' ...
smiData.RigidTransform(108).translation = [-10.1 -9. 1.579707709573533];
smiData.RigidTransform(108).angle = 0;
smiData.RigidTransform(108).axis = [0 0 0];
smiData.RigidTransform(108).ID = ['AssemblyGround[antenkaybetme-1:anten' '-1']'];
smiData.RigidTransform(109).translation = [-7.1981115132472659];
smiData.RigidTransform(109).angle = 1.5725150154632161;
smiData.RigidTransform(109).axis = [-0.99828278658659253]
smiData.RigidTransform(110).translation = [-8.0712090624 2.09960619
smiData.RigidTransform(110).angle = 2.4012955002982608;
smiData.RigidTransform(110).axis = [-0.38803409970208952 0.651701441408713]
smiData.RigidTransform(110).ID = ['AssemblyGround[antenkaybetme-1:distan1-1]'];
smiData.RigidTransform(111).translation = [0 0 0];
smiData.RigidTransform(111).angle = 0;
smiData.RigidTransform(111).axis = [0 0 0];
smiData.RigidTransform(111).ID = 'RootGround[ustcerceve-1]';
smiData.Solid = struct;
smiData.Solid(1).mass = 0.017318376753205248;
smiData.Solid(1).CoM = [-5.9859699064343745 -0.14631803048705072 1];
smiData.Solid(1).MoI = [0.012538275480434142 0.028726038006825241]
smiData.Solid(1).PoI = [0 0 2.6316586398941764E-6];
smiData.Solid(1).color = [0.792156862745098 0.81960784313725488]
smiData.Solid(1).opacity = 1;
smiData.Solid(1).ID = 'Hitec HS-311*:*Varsayilan';
smiData.Solid(2).mass = 0.0010803344938904137;
smiData.Solid(2).CoM = [12.019275112145113 -15.811454441874975]
smiData.Solid(2).MoI = [0.0028089446473626187 0.00031913718373844166]
smiData.Solid(2).PoI = [-1.3870959133172241E-10 -3.4098727606314968E-11]
smiData.Solid(2).color = [0.792156862745098 0.81960784313725488 ]
smiData.Solid(2).opacity = 1;
smiData.Solid(2).ID = 'bacakapartat*:*Varsayilan';
smiData.Solid(3).mass = 0.0015465335201610888;
smiData.Solid(3).CoM = [16.369179154956534 -20.492783972464043]
smiData.Solid(3).MoI = [0.0033765155688838754 0.0018409190343616291]
smiData.Solid(3).PoI = [3.9720292743527317E-9 -1.5468973094682487E-10 ]
smiData.Solid(3).color = [0.792156862745098 0.81960784313725488]

```

```

smiData.Solid(3).opacity = 1;
smiData.Solid(3).ID = 'HSaparat*:*Varsayilan';
smiData.Solid(4).mass = 0.00099274327853437474;
smiData.Solid(4).CoM = [-3.9734972631396173 2.2646579095029247
smiData.Solid(4).MoI = [0.0053723874361411607 0.0053723874361411607 ]
smiData.Solid(4).PoI = [0 0 0];
smiData.Solid(4).color = [0.792156862745098 0.81960784313725488]
smiData.Solid(4).opacity = 1;
smiData.Solid(4).ID = 'distan2*:*Varsayilan';
smiData.Solid(5).mass = 0.00074141586624720086;
smiData.Solid(5).CoM = [-4.766584600909515 2.8535714285714291
smiData.Solid(5).MoI = [0.00398671050365668 0.00398671050365668 ]
smiData.Solid(5).PoI = [0 0 0];
smiData.Solid(5).color = [0.792156862745098 0.81960784313725488 ]
smiData.Solid(5).opacity = 1;
smiData.Solid(5).ID = 'distan4*:*Varsayilan';
smiData.Solid(6).mass = 0.0019601768355735688;
smiData.Solid(6).CoM = [-1.1531856354894063 0.37524294488147369 ]
smiData.Solid(6).MoI = [0.011823429345029095 0.011823429345029095 ]
smiData.Solid(6).PoI = [0 0 0];
smiData.Solid(6).color = [0.792156862745098 0.81960784313725488 ]
smiData.Solid(6).opacity = 1;
smiData.Solid(6).ID = 'distan5*:*Varsayilan';
smiData.Solid(7).mass = 0.020289349536144117;
smiData.Solid(7).CoM = [0.78266195141421746 10.577947173061151 -]
smiData.Solid(7).MoI = [0.055489713623317061 0.045493119033731495 ]
smiData.Solid(7).PoI = [5.0593238834492458E-12 8.6953218507941116E-12 -]
smiData.Solid(7).color = [0.792156862745098 0.81960784313725488 ]
smiData.Solid(7).opacity = 1;
smiData.Solid(7).ID = 'anten*:*Varsayilan';
smiData.Solid(8).mass = 0.00086707957239078384;
smiData.Solid(8).CoM = [0 1.3235714285714295 4.00000000000000089];
smiData.Solid(8).MoI = [0.0046762974215024131 0.0046762974215024131 ]
smiData.Solid(8).PoI = [0 0 0];
smiData.Solid(8).color = [0.792156862745098 0.81960784313725488 ]
smiData.Solid(8).opacity = 1;
smiData.Solid(8).ID = 'distan3*:*Varsayilan';
smiData.Solid(9).mass = 0.0011184069846779717;
smiData.Solid(9).CoM = [-3.1151560294809446 0.57071428571428684 ]
smiData.Solid(9).MoI = [0.0060759230253688888 0.0060759230253688906 ]
smiData.Solid(9).PoI = [0 0 0];
smiData.Solid(9).color = [0.792156862745098 0.81960784313725488 ]

```



```

smiData.Solid(9).opacity = 1;
smiData.Solid(9).ID = 'distan1*:*Varsayilan';
smiData.Solid(10).mass = 0.39777420490991516;
smiData.Solid(10).CoM = [44.061815602703817 1.5446920218757321
smiData.Solid(10).MoI = [4.9900508322661574 5.8218882150683013
smiData.Solid(10).PoI = [0 0.0024872054539930126 -6.508648534381429E-6];
smiData.Solid(10).color = [0.792156862745098 0.81960784313725488
smiData.Solid(10).opacity = 1;
smiData.Solid(10).ID = 'guc kutusu*:*Varsayilan';
smiData.Solid(11).mass = 0.003330372670019266;
smiData.Solid(11).CoM = [-1.5332663729494405 0.509998715592402 ]
smiData.Solid(11).MoI = [0.0036213544539379508 0.003131025292819941 ]
smiData.Solid(11).PoI = [-0.00012879824446226118 -0.00038783534600829911]
smiData.Solid(11).color = [0.792156862745098 0.81960784313725488]
smiData.Solid(11).opacity = 1;
smiData.Solid(11).ID = 'cameratutck*:*Varsayilan';
smiData.Solid(12).mass = 0.010229512885221991;
smiData.Solid(12).CoM = [-1.4539614206691913 0.70210175848316136]
smiData.Solid(12).MoI = [0.0079890849708624465 0.0079275631358273913
smiData.Solid(12).PoI = [2.0279880536488545E-5 2.9470253653420005E-6 ]
smiData.Solid(12).color = [0.792156862745098 0.81960784313725488 ]
smiData.Solid(12).opacity = 1;
smiData.Solid(12).ID = 'camera*:*Varsayilan';
smiData.Solid(13).mass = 0.0018529147215877351;
smiData.Solid(13).CoM = [-5.0057337114093992 -0.068128226189229119 ]
smiData.Solid(13).MoI = [0.00060310888544739486 0.0011024835023431706]
smiData.Solid(13).PoI = [1.3215298311679077E-10 -1.1912319508772164E-9]
smiData.Solid(13).color = [0.792156862745098 0.81960784313725488]

```

## 10. ÖZGEÇMİŞ

1994 yılında Gümüşhane'nin Şiran ilçesinde doğdum. 2012 yılında Lise eğitimimi Özel Enka Teknoloji Lisesinde Endüstriyel Bakım Onarım bölümünde tamamladım. Eğitim hayatıma ara vermeden 2013 yılında Tokat Gaziosmanpaşa Üniversitesinde Mekatronik Mühendisliği programında Lisans eğitimime başladım ve bölüm ikincisi olarak 2017 yılında mezun oldum. Lisans eğitiminin ardından 2017 yılında yine Gaziosmanpaşa Üniversitesinde yüksek lisans eğitimime başladım. 26 Aralık 2019'da yüksek lisans eğitimimi tamamladım.