# OTONOM KONTROLLÜ AKILLI TEKERLEKLİ SANDALYE TASARIMI

**Halil İbrahim ŞAHİN**

**2021**
**YÜKSEK LİSANS TEZİ**
**BİYOMEDİKAL MÜHENDİSLİĞİ**

**Tez Danışmanı**
**Dr. Öğr. Üyesi Ahmet Reşit KAVSAOĞLU**

# OTONOM KONTROLLÜ
# AKILLI TEKERLEKLİ SANDALYE TASARIMI

**Halil İbrahim ŞAHİN**

**T.C.**
**Karabük Üniversitesi**
**Lisansüstü Eğitim Enstitüsü**
**Biyomedikal Mühendisliği Anabilim Dalında**
**Yüksek Lisans Tezi**
**Olarak Hazırlanmıştır**

**Tez Danışmanı**
**Dr. Öğr. Üyesi Ahmet Reşit KAVSAOĞLU**

**KARABÜK**
**Ocak 2021**

Halil İbrahim ŞAHİN tarafından hazırlanan "OTONOM KONTROLLÜ AKILLI TEKERLEKLİ SANDALYE TASARIMI" başlıklı bu tezin Yüksek Lisans Tezi olarak uygun olduğunu onaylarım.

Dr. Öğr. Üyesi Ahmet Reşit KAVSAOĞLU .....
Tez Danışmanı, Biyomedikal Mühendisliği Anabilim Dalı

Bu çalışma, jürimiz tarafından oy birliği ile Biyomedikal Mühendisliği Anabilim Dalında Yüksek Lisans tezi olarak kabul edilmiştir. 21/01/2021

Unvanı, Adı SOYADI (Kurumu) İmzası

Başkan : Dr. Öğr. Üyesi Sabri ALTUNKAYA (NEÜ) .....

Üye : Doç. Dr. Mehmet Akif ERDEN (KBÜ) .....

Üye : Dr. Öğr. Üyesi Ahmet Reşit KAVSAOĞLU (KBÜ) .....

KBÜ Lisansüstü Eğitim Enstitüsü Yönetim Kurulu, bu tez ile Yüksek Lisans derecesini onamıştır.

Prof. Dr. Hasan SOLMAZ .....
Lisansüstü Eğitim Enstitüsü Müdürü

*"Bu tezdeki tüm bilgilerin akademik kurallara ve etik ilkelere uygun olarak elde edildiğini ve sunulduğunu; ayrıca bu kuralların ve ilkelerin gerektirdiği şekilde, bu çalışmadan kaynaklanmayan bütün atıfları yaptığımı beyan ederim."*

Halil İbrahim ŞAHİN

# ÖZET

**Yüksek Lisans Tezi**

**OTONOM KONTROLLÜ**
**AKILLI TEKERLEKLİ SANDALYE TASARIMI**

**Halil İbrahim ŞAHİN**

**Karabük Üniversitesi**
**Lisansüstü Eğitim Enstitüsü**
**Biyomedikal Mühendisliği Anabilim Dalı**

**Tez Danışmanı:**
**Dr. Öğr. Üyesi Ahmet Reşit KAVSAOĞLU**
**Ocak 2021, 97 sayfa**

Yürüme engelli bireyler geçirdikleri bir kaza sonucunda veya doğuştan gelen bir anomalite sonucunda yürüme yeteneklerinin bir kısmını veya tamamını kaybeden kişilerdir. Kuadripleji veya tetrapleji hastaları ise motor nöronlarındaki veya kaslarındaki hasarlanmalardan ötürü kol ve bacak kaslarını hareket ettirememektedir. Bu bireyler, hareket edebilmek için çeşitli biyomedikal cihazlardan yardım alarak hayatlarını normal şekilde devam ettirebilmektedirler.

Temel olarak yürüme engelliler için üretilen akülü araçların bir kumanda konsolu bulunmakta ve bu konsol üzerinde var olan bir yönetme kolu ile aracın ileri, geri, sağ, sol hareketlerini yapması sağlanmaktadır. Ancak, bu yönetme kolunu kontrol edemeyecek derecede felçli hastaların akülü tekerlekli sandalye kullanabilmeleri neredeyse imkânsızdır. Bu aşamada, bir sistemin dışarıdan müdahaleye gerek duymaksızın kendiliğinden çalışabilmesi anlamına gelen otonom kontrolün akülü

tekerlekli sandalyelere kazandırılması gerekir. Otonom kontrol, düşünme işlemini yerine getirecek bir bilgisayara ihtiyaç duyar. Akülü tekerlekli sandalyelere bilgisayar desteği eklenmesiyle elde edilen yeni sistemlere akıllı tekerlekli sandalye denilmektedir. Ortamın önceden haritalanması ve uygun yol bulma algoritmalarının kullanılmasıyla, bireylerin tam olarak bilmedikleri bir ortamdaki istedikleri konuma akıllı tekerlekli sandalye üzerinden ulaşabilmeleri sağlanabilmektedir.

Bu tez çalışması ile hem kapalı mekânlarda hem de açık yerlerde istenilen bir konuma ulaşmayı sağlayan otonom kontrollü akıllı tekerlekli sandalye tasarımı gerçekleştirilmiştir. Tasarlanan bu sistem ile bireylerin rahatlıkla iç mekânlarda dolaşabilmeleri ve sosyal ortamlara uyum sağlayabilmeleri kolaylaştırılmaktadır. Ayrıca her türden akülü tekerlekli sandalyeye uyarlanabilen bir sistem ortaya konulmuştur.

**Anahtar Sözcükler :** Akıllı Tekerlekli Sandalye, Zeki Sistem, Simülasyon, En Kısa Yol Bulma Algoritması, Bulanık Mantık Denetleyicisi, Otonom Kontrol.

**Bilim Kodu** **:** 92504

## ABSTRACT

**M.Sc. Thesis**

**THE DESIGN OF A SMART WHEELCHAIR WITH AUTONOMOUS CONTROL**

**Halil İbrahim ŞAHİN**

**Karabük University**
**Institute of Graduate Programs**
**Department of Biomedical Engineering**

**Thesis Advisor:**
**Assist. Prof. Dr. Ahmet Reşit KAVSAOĞLU**
**January 2021, 97 pages**

Individuals with disabilities are those who have lost some or all their ability to walk because of an accident or a congenital anomaly. Quadriplegia or tetraplegia patients cannot move their arm and leg muscles due to damage in motor neurons or muscles. These individuals can continue their lives normally by getting help from various biomedical devices in order to be able to move.

Essentially for the disabled people, battery powered wheelchairs have a control console and a joystick that allows to move forward, backward, right and left. However, it is almost impossible for patients who are paralyzed to control this joystick for using a powered wheelchair. At this stage, autonomous control, which means that a system can operate automatically without the need for outside intervention, should be brought to powered wheelchairs. Autonomous control requires a computer to perform the thinking process. The new systems obtained by adding computer support to the

powered wheelchairs are called intelligent or smart wheelchairs. By using the pre-mapping of the environment and the use of appropriate route finding algorithms, it is possible for individuals to reach their desired location in an environment that they do not know exactly, via smart wheelchairs.

In this study, autonomous controlled smart wheelchair is designed which enables to reach a desired position in both indoor places and open places. With this designed system, individuals can be navigated indoors and adapt to social environments easily. In addition, a system that can be adapted to any kind of powered wheelchairs has been introduced.

**Key Words** : Smart Wheelchair, Intelligent System, Simulation, Shortest Path Finding Algorithm, Fuzzy Logic Controller, Autonomous Control.

**Science Code :** 92504

## TEŞEKKÜR

Bu tez çalışmasının her aşamasında ilgi ve desteğini esirgemeyen sayın danışmanım Dr. Öğr. Üyesi Ahmet Reşit KAVSAOĞLU'na ve eğitim hayatım boyunca üzerimde emeği olan bütün öğretmenlerime ve arkadaşlarıma teşekkürlerimi sunarım.

Her zaman maddi ve manevi hiçbir desteğini esirgemeyen ve benimle ilgilenerek bu günlere gelmemde büyük katkıda bulunan sevgili aileme teşekkür ederim.

Bu tez çalışması Karabük Üniversitesi Bilimsel Araştırma Projeleri Komisyonu tarafından FYL-2019-2040 numaralı proje kapsamında desteklenmiştir.

# İÇİNDEKİLER

**Sayfa**

KABUL ..... ii
ÖZET ..... iv
ABSTRACT ..... vi
TEŞEKKÜR ..... viii
İÇİNDEKİLER ..... ix
ŞEKİLLER DİZİNİ ..... xiii
ÇİZELGELER DİZİNİ ..... xvi
SİMGELER VE KISALTMALAR DİZİNİ ..... xvii

BÖLÜM 1 ..... 1
GİRİŞ ..... 1

BÖLÜM 2 ..... 7
MATERYAL VE YÖNTEM ..... 7
2.1 KULLANICI İLE ETKİLEŞİM BİRİMİ ..... 7
2.1.1 Dokunmatik Ekran ..... 8
2.1.2 Mikrofonlu Kulaklık ..... 11
2.2 ÇEVRE HAKKINDA BİLGİ ALMA BİRİMİ ..... 12
2.2.1 Derinlik Algılayıcı Kamera ..... 12
2.3 KARAR VERME BİRİMİ ..... 15
2.3.1 Tek Kart Bilgisayar ..... 15
2.3.2 Tek Kart Mikrodenetleyici ..... 16
2.4 SİNYALİZASYON BİRİMİ ..... 19
2.4.1 LED Aydınlatmalar ..... 20
2.4.2 Vızlayıcı ..... 21
2.5 HAREKET KONTROL BİRİMİ ..... 22
2.5.1 DA Motor Sürücüsü ..... 22
2.5.2 Hız Sensörü ..... 23

**Sayfa**

BÖLÜM 3 ..... 25
SİMÜLASYON ORTAMININ HAZIRLANMASI ..... 25
3.1 SİMÜLASYON ORTAMINA GENEL BAKIŞ ..... 25
3.2 SAHNE NESNELERİ ..... 27
3.2.1 Eklemler (Joints) ..... 28
3.2.2 Şekiller (Shapes) ..... 28
3.2.3 Yakınlık Sensörleri (Proximity Sensors) ..... 29
3.2.4 Görüntü Sensörleri (Vision Sensors) ..... 29
3.2.5 Kuvvet Sensörleri (Force Sensors) ..... 30
3.2.6 Grafikler (Graphs) ..... 30
3.2.7 Kameralar (Cameras) ..... 30
3.2.8 Işıklar (Lights) ..... 31
3.2.9 Yollar (Paths) ..... 31
3.2.10 Kuklalar (Dummies) ..... 31
3.2.11 Sekizli Ağaçlar (OC Trees) ..... 31
3.2.12 Nokta Bulutları (Point Clouds) ..... 32
3.3 HESAPLAMA MODÜLLERİ ..... 32
3.3.1 Kinematik Modülü ..... 32
3.3.2 Dinamik Modülü ..... 33
3.3.3 Çarpışma Algılama Modülü ..... 33
3.3.4 Mesafe Hesaplama Modülü ..... 33
3.3.5 Yol Planlama Modülü ..... 33
3.4 SİMÜLASYON KONTROL MEKANİZMALARI ..... 34
3.4.1 Gömülü Komut Dosyaları (Embedded Scripts) ..... 36
3.4.2 Uzantılar (Add-ons) ..... 37
3.4.3 Eklentiler (Plug-ins) ..... 37
3.4.4 Uzak API İstemcileri (Remote API Clients) ..... 38
3.4.5 ROS Düğümleri (ROS Nodes) ..... 38
3.4.6 BlueZero Düğümleri (BlueZero Nodes) ..... 38
3.5 KOMUT DOSYALARI ..... 39
3.5.1 Ana Komut Dosyası (Main Script) ..... 39
3.5.1.1 Başlatma Fonksiyonu (Initialization Fuction) ..... 40

**Sayfa**

3.5.1.2 Hareket Ettirme Fonksiyonu (Actuation Function) ..... 40
3.5.1.3 Algılama Fonksiyonu (Sensing Function) ..... 40
3.5.1.4 Restorasyon Fonksiyonu (Restoration Function)..... 41
3.5.2 Alt Komut Dosyaları (Child Scripts)..... 41
3.5.2.1 İş Parçacıklı Olmayan Alt Komut Dosyaları (Non-Threaded Child Scripts) ..... 41
3.5.2.2 İş Parçacıklı Alt Komut Dosyaları (Threaded Child Scripts) ..... 41
3.6 SİMÜLASYON SÜRECİ..... 42
3.7 NESNE MODELLERİNİN EKLENMESİ ..... 44
3.7.1 Tekerlekli Sandalye ..... 44
3.7.2 Kat Planı ..... 45
3.8 SİMÜLASYON MODELİNİN ARAYÜZ YAZILIMI İLE ETKİLEŞİME GEÇEREK KULLANILMASI..... 46

BÖLÜM 4 ..... 48
YOL BULMA ALGORİTMASININ BELİRLENMESİ ..... 48
4.1 DÜĞÜM İLE KOMŞULARI ARASINDAKİ UZAKLIK ÖLÇÜTLERİ ..... 49
4.1.1 Öklid Uzaklığı (Euclidean Distance)..... 50
4.1.2 Şehir Bloğu Uzaklığı (City Block Distance) ..... 50
4.1.3 Satranç Tahtası Uzaklığı (Chessboard Distance) ..... 51
4.2 A* ALGORİTMASI..... 53
4.3 ATLAMA NOKTASI ARAMA ..... 57

BÖLÜM 5 ..... 62
ARAYÜZ VE MODELLEME..... 62
5.1 BULANIK MANTIK..... 70
5.2 BULANIK MANTIK DENETLEYİCİSİ ..... 71
5.2.1 Mamdani Modeli ..... 71
5.2.2 Sugeno Modeli..... 73
5.3 TEKERLEKLİ SANDALYENİN KİNEMATİK MODELİ..... 77
5.4 ÜYELİK FONKSİYONLARI..... 79
5.5 BULANIK KURAL LİSTESİ..... 83

**Sayfa**

BÖLÜM 6 ..... 85
SONUÇLAR VE ÖNERİLER ..... 85
KAYNAKLAR ..... 87

ÖZGEÇMİŞ ..... 97

## ŞEKİLLER DİZİNİ

**Sayfa**


Şekil 1.1. Otonom kontrollü akıllı tekerlekli sandalye sisteminin blok şeması. ..... 2

Şekil 2.1. Asus VT168H dokunmatik ekran..... 8

Şekil 2.2. Bluetooth destekli kablosuz mikrofonlu kulaklık. ..... 11

Şekil 2.3. Bluetooth ve Wi-Fi adaptörü..... 12

Şekil 2.4. Akıllı tekerlekli sandalye sisteminin sol ön tarafına takılan derinlik algılayıcı kamera. ..... 13

Şekil 2.5 Intel RealSense D435 renkli yayın akışından örnek bir görüntü. ..... 14

Şekil 2.6. Intel RealSense D435 kızılötesi yayın akışından örnek bir görüntü. ..... 14

Şekil 2.7. Intel RealSense D435 derinlik yayın akışından örnek bir görüntü. ..... 14

Şekil 2.8. UPBoard tek kart bilgisayarı. ..... 15

Şekil 2.9. ATmega328 mikrodenetleyicisinin dâhilî blok diyagramı..... 18

Şekil 2.10. ATmega32U4 mikrodenetleyicisinin dâhilî blok diyagramı..... 19

Şekil 2.11. Akıllı tekerlekli sandalye sisteminin hareket yönleriyle ilişkilendirilen LED aydınlatmalar. ..... 20

Şekil 2.12. Hazırlanan akıllı tekerlekli sandalye sisteminin önden (a) ve arkadan (b) görünümü. ..... 21

Şekil 2.13. Vızlayıcı. ..... 21

Şekil 2.14. Çift DA Motor Sürücü Kartı. ..... 22

Şekil 2.15. Hall etkili manyetik sensör ve tekere takılan mıknatıslara verdiği tepki. ..... 24

Şekil 3.1. CoppeliaSim yazılımının sahne nesneleri. ..... 28

Şekil 3.2. CoppeliaSim yazılımının kontrol mimarisini gösteren şematik diyagram. ..... 35

Şekil 3.3. Ana simülasyon döngüsü. ..... 43

Şekil 3.4. Gerçek zamanlı simülasyon döngüsü. ..... 43

Şekil 3.5. Simülasyonda kullanılan sıradan bir tekerlekli sandalye modeli. ..... 45

Şekil 3.6. KBÜ Mühendislik Fakültesi binasının dördüncü katının modellenmesi. ..... 45

Şekil 3.7. CoppeliaSim yazılımında simülasyonun çalıştırılması. ..... 47

Şekil 4.1. Bir düğümün (node) komşuları. a) Dörtlü komşulukta bir düğümün ($n$) komşuları (•). b) Sekizli komşulukta bir düğümün ($n$) komşuları (•). .... 49

**Sayfa**

Şekil 4.2. Kare hücreli ızgaradaki bir düğüm için Öklid mesafeleri. a) Geçerli *n* düğümü ve sekizli komşuluktaki komşuları. b) Komşu düğümlerin geçerli *n* düğümüne olan Öklid mesafe değerleri. ..... 50

Şekil 4.3. Kare hücreli ızgaradaki bir düğüm için şehir bloğu mesafeleri. a) Geçerli *n* düğümü ve sekizli komşuluktaki komşuları. b) Komşu düğümlerin geçerli *n* düğümüne olan şehir bloğu mesafe değerleri. ..... 51

Şekil 4.4. Kare hücreli ızgaradaki bir düğüm için satranç tahtası mesafeleri. a) Geçerli *n* düğümü ve sekizli komşuluktaki komşuları. b) Komşu düğümlerin geçerli *n* düğümüne olan satranç tahtası mesafe değerleri. ..... 52

Şekil 4.5. Koordinat ekseninde Öklid, şehir bloğu ve satranç tahtası ölçütüne göre orijinden *r* uzaklıktaki noktalar. ..... 52

Şekil 4.6. A* algoritmasının akış diyagramı. ..... 56

Şekil 4.7. Kare hücreli 2 × 5 boyutlu ızgaranın çapraz köşelerinde bulunan bir başlangıç düğümünden bir bitiş düğümüne eşdeğer yollar. ..... 57

Şekil 4.8. Atlama noktası aramasında çapraz yönde (düz çizgi) ilerlenirken işlenen yönler (noktalı çizgili) ve ilk önce işlenen düz yönler (kesikli çizgili). ..... 58

Şekil 4.9. Genişletilen geçerli *n* düğümü için komşularının budanması. a) Düz yönde budama. b) Çapraz yönde budama. ..... 59

Şekil 4.10. Genişletilen geçerli *n* düğümü için zorunlu komşular. a) Düz yönde zorunlu komşu. b) Çapraz yönde zorunlu komşu. ..... 60

Şekil 4.11. A* algoritmasında (a) ve atlama noktası aramasında (b) oluşturulan düğümlerin kıyaslanması. ..... 60

Şekil 5.1. Arayüzde ayarların girilmesi. ..... 63

Şekil 5.2. Arayüzde güzergâh belirlemesi örneği. ..... 65

Şekil 5.3. Kat planının 20 cm × 20 cm hücreli ızgarada temsilî. ..... 66

Şekil 5.4. Kat planının 20 cm × 20 cm hücreli ızgarada temsilî (yakın görünüm). ..... 67

Şekil 5.5. Izgara üzerine oturtulmuş kat planının pikselize edilmesi. ..... 67

Şekil 5.6. Izgara üzerine oturtulmuş kat planının pikselize edilmesi (yakın görünüm). ..... 68

Şekil 5.7. Mamdani modeline göre bulanık mantık denetleyicisi. ..... 72

Şekil 5.8. Sugeno modeline göre bulanık mantık denetleyicisi. ..... 74

Şekil 5.9. Tekerlekli sandalyenin kinematik modeli. ..... 77

Şekil 5.10. Tekerlekli sandalyenin güzergâh takibinde kullanılan bulanık mantık kontrol sisteminin blok diyagramı. ..... 79

Şekil 5.11. En sık kullanılan bulanık küme biçimleri. a) Üçgensel b) Yamuksal c) Normal dağılım d) Tekton. ..... 80

**Sayfa**

Şekil 5.12. Mesafe için giriş üyelik fonksiyonları..... 81
Şekil 5.13. Açı için giriş üyelik fonksiyonları..... 82
Şekil 5.14. Hız için Mamdani modelindeki eşdeğer üyelik fonksiyonları. ..... 82

# ÇİZELGELER DİZİNİ

**Sayfa**

Çizelge 2.1. Intel RealSense D435 teknik özellikleri. ..... 13
Çizelge 2.2. UPBoard teknik özellikleri. ..... 16
Çizelge 5.1. Bulanık mantıktaki temel işleçler. ..... 71
Çizelge 5.2. Dilsel çıkarım tablosu. ..... 83

## SİMGELER VE KISALTMALAR DİZİNİ

### SİMGELER

$d$ : mesafe fonksiyonu

$f$ : toplam maliyet fonksiyonu

$g$ : üretim maliyeti fonksiyonu

$h$ : sezgisel maliyet fonksiyonu

$p$ : ebeveyn düğüm fonksiyonu

$V$ : doğrusal hız

$T$ : örnekleme zamanı

$\theta$ : cismin hareket yönü ile apsisler ekseni arasındaki açı

$\mu$ : üyelik fonksiyonu

$\sigma$ : ölçek parametresi

$\varphi$ : cismin hareket yönü ile cisim ve hedefinden geçen doğrultu arasındaki açı

$\omega$ : açısal hız

**KISALTMALAR**

| | |
|---|---|
| 3B | : Üç Boyutlu |
| API | : Application Programming Interface (Uygulama Programlama Arayüzü) |
| BFS | : Breadth-First Search (Sığ Öncelikli Arama) |
| BØ | : BlueZero |
| CAD | : Computer-Aided Design (Bilgisayar Destekli Tasarım) |
| CPU | : Central Processing Unit (Merkezî İşlem Birimi) |
| DA | : Doğru Akım (Direct Current, DC) |
| DFS | : Depth-First Search (Derinlik Öncelikli Arama) |
| DGA | : Darbe Genişlik Ayarlı (Pulse-Width Modulation, PWM) |
| DoF | : Degree of Freedom (Serbestlik Derecesi) |
| EAN | : European Article Number (Avrupa Madde Numarası) |
| EEPROM | : Electrically Erasable Programmable Read-Only Memory (Elektriksel Silinebilir Programlanabilir Salt Okunur Bellek) |
| EPROM | : Erasable Programmable Read-Only Memory (Silinebilir Programlanabilir Salt okunur Bellek) |
| GPS | : Global Positioning System (Küresel Konumlandırma Sistemi) |
| IDE | : Integrated Development Environment (Tümleşik Geliştirme Ortamı) |
| IDT | : Interdigital Transducer (İnterdijital Transdüser) |
| JPS | : Jump Point Search (Atlama Noktası Arama) |
| LED | : Light Emitting Diode (Işık Yayan Diyot) |
| MER | : Mars Exploration Rover (Mars Keşif Aracı) |
| NASA | : U.S.A. National Aeronautics and Space Administration (A.B.D. Ulusal Havacılık ve Uzay Dairesi) |
| OpenGL | : Open Graphics Library (Açık Grafik Kütüphanesi) |
| PDF417 | : Portable Data File 417 (Taşınabilir Veri Dosyası 417) |
| QR | : Quick Response (Çabuk Tepki) |
| RAM | : Random Access Memory (Rastgele Erişim Belleği) |
| ROS | : Robot Operating System (Robot İşletim Sistemi) |
| RRT | : Rapidly-exploring Random Tree (Keşfeden Rastgele Ağaçlar) |
| UPC | : Universal Product Code (Evrensel Ürün Kodu) |
| TCP | : Transmission Control Protocol (İletim Kontrol Protokolü) |

V-REP : Virtual Robot Experimentation Platform (Sanal Robot Deney Platformu)

Wi-Fi : Wireless Fidelity (Kablosuz Bağlantı)

ØMQ : Zero Message Queue (Sıfır Mesaj Kuyruğu)

# BÖLÜM 1

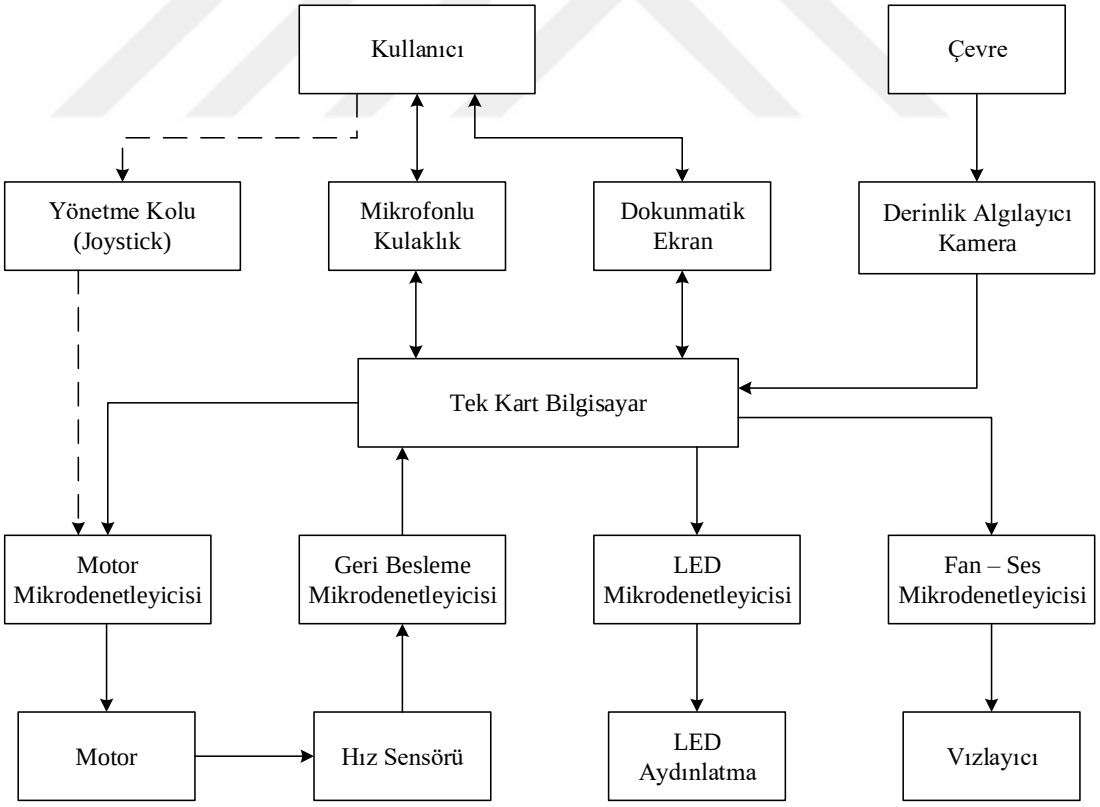
# GİRİŞ

Orta derecede engelli bireyler, tekerlekli sandalyeleri bağımsız olarak kullanarak temel ihtiyaçlarını karşılayabilmektedir. Diğer taraftan ileri derecede engelli bireylerde, tekerlekli sandalyelerin bağımsız olarak kullanılması zor ya da imkânsızdır. Özellikle hem alt ekstremiteleri hem de üst ekstremiteleri kullanamayan engelli bireylerin tekerlekli sandalye ile hareket kabiliyeti bir başkasının yardımına bağlıdır. Gün geçtikçe, teknolojik gelişmelerle uyumlu yapılaşmaların da artması sonucunda bu gibi durumlar için hem bina içerisinde hem de sokakta tekerlekli sandalye kullanımını kolaylaştıracak bir sistemin tasarlanması gereksinim hâline gelmiştir.

Engelli bireyler, hayatları boyunca kullanmaları gereken tekerlekli sandalyeler için kullanım biçimini kolaylaştırıcı yeniliklere ihtiyaç duymaktadır. Bu aşamada devreye giren akıllı tekerlekli sandalyeler, tüm dünyada ilgi gören otonom mobil robotların önemli bir alt sınıfını temsil etmektedir. Avrupa Birliği'nde yarı otonom veya tam otonom akıllı tekerlekli sandalyelerden yararlanabilecek iki milyondan fazla kısmi engelli birey bulunmaktadır. Avrupa kurumları, üniversiteleri ve şirketleri, çeşitli seviyelerde navigasyon zekâsına sahip yarı otonom ve tam otonom tekerlekli sandalyelerin geliştirilmesi yönünde yoğun bir çaba göstermektedir [1].

Tekerlekli sandalyeyi kendi başına hareket ettirmekte zorlanan kullanıcılar için akülü (elektrikli) tekerlekli sandalyeler ideal çözüm sunmaktadır. Akülü tekerlekli sandalyeler, genel anlamda tekerlek kombinasyonuna ve motor gücüne bağlı olarak sınıflandırılmaktadır. Akülü tekerlekli sandalye, tüm tekerlekleri katı lastiğe sahipse iç mekân, ön tekerlekleri katı lastiğe arka tekerlekleri havalı lastiğe sahipse iç/dış mekân ve tüm tekerlekleri havalı lastiğe sahipse dış/iç mekân veya dış mekân olarak adlandırılır [2].

Bu tez çalışması kapsamında, bir dış/iç mekân akülü tekerlekli sandalye, otonom kontrollü akıllı tekerlekli sandalye sistemine dönüştürülmüştür. Otonom kontrol algoritmalarının çalışacağı ve sensörlerden gelen bilgilerin işleneceği bir bilgisayar yazılımı oluşturulmuştur. Tekerlekli sandalyenin gideceği konum bilgisi, ekrana yansıtılan arayüze kullanıcının dokunmasıyla elde edilmektedir. Konum bilgileri doğrultusunda, bilgisayar üzerinde çalışan yazılım aracılığıyla en kısa yol bulunmaktadır. Ön tarafa yerleştirilen derinlik algılayıcı kameradan alınan yol görüntüleri işlenmekte ve uygun hareket komutları tekerlekli sandalyenin motor sürücü devresine gönderilmektedir. Ortamdaki hareket hâlinde bulunan kişi sayısının artması durumunda, karşılaşılabilecek risklere karşı çarpışma önleyici komutlar eklenmiştir. Akıllı tekerlekli sandalyenin önüne beklenmedik bir engelin çıkması durumunda, sistemin durdurularak sesli ikaz vermesi ve alternatif bir yol üzerinden hareketine devam etmesi sağlanmaktadır. Otonom kontrollü akıllı tekerlekli sandalye sisteminin blok şeması Şekil 1.1'de gösterilmiştir.




Şekil 1.1. Otonom kontrollü akıllı tekerlekli sandalye sisteminin blok şeması.

Otonom kontrollü akıllı tekerlekli sandalye sistemindeki dokunmatik ekran ve mikrofonlu kulaklık hem kullanıcıdan komut alınması hem de kullanıcının uyarılması için kullanılmaktadır. Mikrofonlu kulaklık ile kullanıcının ses komutları tek kart bilgisayara aktarılmakta ve sistemin sesli uyarıları kullanıcıya iletilmektedir. Dokunmatik ekrana yansıtılan görsel arayüz üzerinden hem kullanıcının istikamet bilgisini sisteme girebilmesi hem de sisteme dair bilgilerin kullanıcıya sunulması sağlanmaktadır.

Akıllı tekerlekli sandalyenin bulunduğu ortamda yer alan çevre elemanlarının tespiti için derinlik algılayıcı kamera kullanılmaktadır. Ayrıca kameranın aldığı görüntüler işlenerek yakındaki QR (Quick Response, Çabuk Tepki) kodları okunabilmektedir. Bu sayede, duvar ve kapı gibi çeşitli mekâna bağlı unsurlara QR kodlu etiketler konulabilmekte ve akıllı tekerlekli sandalye sisteminin konum bilgisi güncellenebilmektedir. QR kodlarının içeriğinin genişletilebilir olması, akıllı tekerlekli sandalye sistemi için yardımcı farklı işaret ve uyarıların ortama kolayca yerleştirilebilmesini sağlamaktadır.

Akıllı tekerlekli sandalye sisteminin ortam haritası üzerinden konumlandırma yapabilmesi, istikamete göre güzergâh belirlemesi, motorlara hareket komutları verilerek sistemin yönlendirilmesi, derinlik algılayıcı kameradan gelen bilgilere göre çevre unsurlarının (duvar ve kapı gibi) tespit edilmesi ve karşılaşılan engellerden kaçınma işlemleri için tek kart bilgisayar kullanılmaktadır. Ayrıca bu tek kart bilgisayarın işlem yükünün hafifletilebilmesi ve akıllı tekerlekli sandalye sisteminin daha hızlı tepki verebilmesi için yardımcı olarak dört adet mikrodenetleyici sisteme dâhil edilmiştir.

Akıllı tekerlekli sandalyenin hareket ettiği yönleri çevresine bildirebilmesi için günümüz kara taşıtlarında yer alan ileri, geri, sağ, sol, durma ve aydınlatma lambaları ile sinyalizasyon sistemi oluşturulmuştur. Bu sinyalizasyon sistemine ek olarak hem geri harekette hem de öne çıkan engellerde uyarı verecek sesli ikaz için vızlayıcı (buzzer) yerleştirilmiştir.

Akıllı tekerlekli sandalye sisteminin hareket etmesini sağlayan iki adet DA (Doğru Akım) motorun sürülebilmesi için çift DA motor sürücü kartı kullanılmaktadır. Çift DA motor sürücü kartı ile sürülen motorların katettiği mesafeyi belirleyen hız sensörleri ile akıllı tekerlekli sandalye sisteminin daha hassas komunlandırılması sağlanmaktadır.

Akıllı tekerlekli sandalye sisteminin test edilmesi için kullanılan ortamın haritalanması yapılmıştır. Uygun bir ölçekte küçültülmüş temsilî harita hazırlanırken bir tekerlekli sandalyenin geçebileceği genişliğe sahip tüm yollar belirlenmiştir. Akıllı tekerlekli sandalyelerin bir konumdan diğerine geçişi esnasında en kısa yol bulma algoritmaları uygulanırken acil durumlarda takip edilebilecek alternatif güzergâhların da hesaplanabiliyor olması gerekmektedir [3]. Otonom kontrol sağlayan yazılım hazırlanırken acil durumlar göz önünde bulundurulmuş ve aniden engel çıkması hâlinde sistemin durdurulmasını sağlayan acil durdurma butonları (fiziksel durdurma butonu ve yazılım arayüzündeki durdurma butonu) sisteme eklenmiştir.

Akıllı tekerlekli sandalye üzerine yapılan araştırmaların genel amacı, akülü tekerlekli sandalyeye yardımcı teknolojiler eklenerek zeki sistem özelliği kazandırılmasıdır [4]. Richard C. Simpson tarafından yayınlanan incelemeye göre, bir akıllı tekerlekli sandalye sistemi tasarımında kullanılabilecek farklı tekerlekli sandalye modelleri bulunmaktadır [5]. Geliştirilen akıllı tekerlekli sandalyelerin büyük çoğunluğu, piyasadaki tekerlekli sandalyelerin üzerinde değişiklikler yapılarak üretilmiştir. İlk üretilen akıllı tekerlekli sandalyeler, üzerine koltuklar eklenen mobil robotlar şeklindedir [6,7]. Sonraki akıllı tekerlekli sandalye sistemleri ise akülü tekerlekli sandalyelere takılıp çıkarılabilen eklentiler olarak tasarlanmıştır [8]. Tüm bu tasarımlar, tekerlekli sandalyelerin kullanılma biçimini kolaylaştırma ve engellere çarpmasını önleme gibi ortak noktalara sahiptir.

Literatürde yer alan Hephaestus ve NavChair gibi akıllı tekerlekli sandalyelerde engelden kaçınma, kapı geçişi ve otomatik duvar takibi gibi çalışma yöntemleri bulunmaktadır [9,10]. Juan C. G. Garcia ve arkadaşları, tekerlekli sandalye sisteminin bulunduğu yer ve koordinat bilgisinin elde edilmesi için kâğıda çıktı alınarak düşük maliyette üretilebilen yapay yer işaretleri kullanmaktadır [11].

Mevcut akıllı tekerlekli sandalye sisteminde kullanılabilecek yol bulma algoritmaları için yapılan çalışmalardan bazıları şunlardır. Alejandro S. Martinez-Sala ve arkadaşları yönlendirme amacıyla A* algoritmasını kullanarak görme engelli kişiler için iç mekân navigasyon sistemini uygulamıştır [12]. Hong-Mei Zhang ve arkadaşları karmaşık ortamda hareket edecek mobil robotlar için güvenli bir yol tahmini sunabilen geliştirilmiş A* algoritması önermiştir [13]. Davide Aversa ve arkadaşları, atlama noktası aramasına dayalı envanter odaklı yol planlamasını tanıtmıştır [14]. Kaijun Zhou ve arkadaşları, 125 simülasyon deneyi sonucunda, JPS (Jump Point Search, Atlama Noktası Arama) algoritmasının gerçek zamanlı yerel yol planlaması için verimli bir yöntem olduğunu ve yapılandırılmış kentsel ortamda en uygun yolu hızla bulabildiğini göstermiştir [15].

Birçok akademik çalışma, bulanık mantık teorisini mobil robotları kontrol etmek için bir çözüm olarak önermektedir. Malek Njah ve Mohamed Jallouli akülü tekerlekli sandalyenin ultrasonik sensörlerinden alınan verileri bulanık mantık denetleyicisi ile işleyerek engellerden kaçınmıştır [16]. Thoa T. Mac ve arkadaşları, iki tekerlekli robotun hedefe giderken engelden kaçınması için çok girişli çok çıkışlı bulanık mantık denetleyicisi tasarlayarak uygulamasını MATLAB simülasyonu ile incelemiştir [17].

Bu tez çalışması şu şekilde düzenlenmiştir. Birinci bölümde konuya bir giriş yapıldıktan sonra literatürde bu konu üzerinde yapılan çalışmaların birkaçına yer verilmiştir. İkinci bölümde, tez çalışması kapsamında hazırlanan otonom kontrollü akıllı tekerlekli sandalye sisteminde kullanılan materyallerden bahsedilmiştir. Kullanılan materyallerin sistem üzerindeki görevleri ve bu görevlerin bağlantı kurdukları yöntemlere değinilmiştir.

Üçüncü bölümde hazırlanan otonom kontrollü akıllı tekerlekli sandalye sisteminin algoritmalarının test edilmesinde kullanılan CoppeliaSim yazılımı ve sunduğu özellikler irdelenmiştir. Simülasyon süreci anlatılarak, tez çalışmasının test edildiği ortamın üç boyutlu modeli verilmiştir. Dördüncü bölümde, en kısa yol bulma algoritmaları arasındaki temel yaklaşımlardan bahsedilmiştir. Bitişe giden en uygun yol hesaplanırken bir sonraki adımın en düşük uzaklık ağırlığına göre belirlendiği Dijkstra algoritmasının bir iyileştirmesi olan A* algoritmasına değinilmiştir. Bu

algoritmada, yönlendirme tespit edilirken sezgisel fonksiyondan yararlanılmaktadır. Elde edilen sezgisel değerlere dayalı olarak, en uygun yol tahmin edilmektedir.

Dördüncü bölümün devamında atlama noktası aramasının mekanik ayrıntıları ve algoritmik özellikleri açıklanmıştır. Engellerin yoğunluğuna bağlı olarak en uygun yolun bulunması A* algoritmasıyla zaman alıcı olabilmektedir [18]. Buna çözüm olarak geliştirilen atlama noktası araması, tek düzenli (uniform) bir grafik üzerinde yatay, dikey veya çapraz olarak komşu düğümlerin budanması varsayımıyla çalışan A* algoritmasının bir iyileştirmesidir.

Beşinci bölümde, kullanılan yazılım arayüzü ve deney ortamının harita ile temsil edilmesi hakkında bilgiler verilmiştir. Kullanıcı ile görsel bir arayüz üzerinden bilgi alışverişini sağlayan yazılım, akıllı tekerlekli sandalye sistemindeki tüm bileşenlerle olan bağlantıları kurmakta ve algoritmaları işletmektedir. Arayüzün hazırlanmasında kullanılan üçüncü parti yazılım bileşenlerinin çalışmaya kazandırdıkları yetkinliklerden de bahsedilmiştir.

Beşinci bölümün devamında klasik mantıktaki bazı temel kavramlar ve ilkeler tanıtılmıştır. Ardından klasik mantığın, tez çalışmasında kullanılan bulanık mantığa genellemesi incelenmiştir. Otonom kontrollü akıllı tekerlekli sandalyenin bulanık mantık denetleyicisi için tekerlekli sandalyenin kinematik modeli ile teorik zemin oluşturulmuştur.

Altıncı bölümde, hazırlanan otonom kontrollü akıllı tekerlekli sandalye sisteminden çıkarılan sonuçlara ve ilerleyen çalışmalarda üzerine eklenebilecek geliştirmelere değinilmiştir.

# BÖLÜM 2

# MATERYAL VE YÖNTEM

Tez çalışmasının bu bölümü, otonom kontrollü akıllı tekerlekli sandalyenin yönetme kolu kullanılmadan kendiliğinden hareket edebilmesi için gereken donanımsal eklemeleri kapsamaktadır. Tekerlekli sandalye kullanıcısından ekran üzerine dokunma yoluyla alınan istikamet bilgisine göre yardımcı yazılım aracılığıyla güzergâh belirlenebilmektedir. Kamera aracılığıyla çevrenin analiz edilmesi sonucunda konum bilgisi elde edilmekte ve güzergâha göre harekete yön verilmektedir. Motorların sürülmesi, LED (Light Emitting Diode, Işık Yayan Diyot) aydınlatmaların açılıp kapanması ve hız sensöründen veri alınması, tek kart bilgisayarın farklı seri haberleşme bağlantılarına takılan bağımsız mikrodenetleyiciler aracılığıyla yapılmaktadır. Bu sayede, işlem tepki sürelerinin en aza indirilmesi sağlanmaktadır.

## 2.1 KULLANICI İLE ETKİLEŞİM BİRİMİ

İnsan düşündüğü komutları çeşitli organlar yardımıyla dışarıya iletmektedir. Organların hareketleri görüntü işleme, ses işleme ya da dokunma sensörleri yardımıyla komut olarak değerlendirilebilmektedir. Akıllı tekerlekli sandalyelerde, kullanıcı ile etkileşim kurulabilmesi için bu yöntemlerden faydalanılabilmektedir. Bu çalışmada, kullanıcının el hareketleri dokunmatik ekran yardımıyla değerlendirilerek ya da kullanıcının sesi mikrofon yardımıyla değerlendirilerek komutlara dönüştürülmektedir. Ayrıca kullanıcı ekran ile görsel olarak bilgilendirilirken, mikrofonlu kulaklık ile de işitsel olarak bilgilendirilebilmektedir.

### 2.1.1 Dokunmatik Ekran

Dokunmatik ekran, görüntüleme alanındaki bir dokunuşun varlığını ve konumunu algılayabilen elektronik bir görüntü panelidir. Bir dokunuş için konum belirlenirken biri X ekseni ve diğeri Y ekseni üzerinde olan iki ölçüm değeri gereklidir. Dokunmatik ekranlar, ekrana parmak veya el ile dokunularak görüntülenenle doğrudan etkileşime girilmesini sağlar. Böylece, bir fare, iz topu veya dokunmatik yüzey tarafından kontrol edilen bir işaretçiye gerek kalmamaktadır [19].

Dokunmatik ekranın iki temel ilkesi vardır. Dirençli ve kapasitif dokunmatik ekranların bulunduğu birinci ilkede bir kaplama yüzey ile temas kurulur. Yüzey akustik dalgalı ve kızılötesi dokunmatik ekranların bulunduğu ikinci ilkede ise ekran üzerine yansıtılan sinyaller kesintiye uğrar [20]. Bu çalışmada Şekil 2.1'de gösterilen kapasitif dokunmatik ekran tercih edilmiştir.

Şekil 2.1. Asus VT168H dokunmatik ekran.

Dirençli dokunmatik ekranlar birkaç katman malzemeden yapılmıştır. En dışta, içteki iletken malzemelerden yalıtılan yüzey katmanı bulunur. Yüzey katmanının altında, uzunlukları boyunca dirençleri değişen iki büyük şeffaf iletken kaplama vardır [21]. Bu iki ince elektriksel olarak iletken katman, bir dizi çok küçük şeffaf yalıtım noktalarından oluşan küçük bir boşlukla birbirinden ayrılmıştır. Dış yüzeye basıldığında, iç tabakalar birbirine temas eder ve panel bir çift gerilim bölücü görevi görür. Devrenin tamamlanmasıyla, ekrana basılan yeri gösteren elektrik akımları oluşur ve yorumlanması için denetleyiciye gönderilir [19].

Dirençli dokunmatik ekranlar, parmak, tırnak, kalem veya eldiven tarafından uygulanan basıncı tanıyabilir. Uzun süre kullanılmadığı durumlarda hesaplamaların doğruluk seviyesi düşebilmektedir. Dirençli dokunmatik ekranların tepki süresi yavaştır [22]. Etkileşim kurulabilmesi için diğer dokunmatik ekranlara kıyasla daha yüksek bir kuvvet uygulanması gerekir [21]. Buna rağmen, ek katmanlarından dolayı yalnızca %75 optik şeffaflık sunabilmektedir. Toz veya su gibi kirletici maddelerden etkilenmezler. Ancak keskin nesnelerin oluşturduğu çizikler, panel katmanlarına kolayca zarar verebilmektedir [23]. Genel olarak tüm dokunmatik ekran çeşitleri arasında maliyeti en uygun olanıdır. Fabrikaların yanı sıra restoranlar ve hastanelerde sıklıkla kullanılmaktadır [19].

Kapasitif dokunmatik ekranlar, insan vücudunun elektrik iletkeni olması özelliğinden yararlanır. Bu nedenle ekranın yüzeyine dokunmak, ekranın elektrostatik alanında bir bozulmaya neden olur ve kapasitanstaki değişiklik olarak ölçülebilir [19]. Kapasitif dokunmatik ekran paneli, elektrik yüklerini depolayan şeffaf bir iletken malzeme ile kaplanmıştır [23]. Panele dokunulduğunda, temas noktasından az miktarda yük kullanıcıya aktarılır. Temas noktasının konumu ise, ekranın her bir köşesinde algılanan yükteki göreceli farklılıkların karşılaştırılmasıyla hesaplanır [24]. Dokunma yerini daha kesin bir şekilde algılama ve dokunan nesnenin alanı ile şeklini algılama yetenekleri, ekran yüzeyine birden çok kapasitif sensör yerleştirilerek elde edilebilmektedir [21].

Kapasitif dokunmatik ekranlar, görüntünün yaklaşık %90 kadarını geçirir ve yalnızca elektrik ileten nesnelere tepki verir [24]. Dirençli dokunmatik ekranların aksine, sıradan kalem veya eldiven gibi elektriksel olarak yalıtkan bir malzemeyle kullanılamazlar. Bunun yerine statik elektrik üreten özel bir kapasitif kalem veya parmak uçlu eldiven yardımıyla kullanılabilirler [19]. Yalnızca parmak veya el ile dokunulması gerektiğinden üzerine basılan bir noktanın kesin doğrulukta belirlenmesi biraz zordur. Kapasitif dokunmatik ekran güçlü bir dayanıklılığa sahiptir. Dokunmayı sürekli olarak algılayabilirken uzun süre kullanılmadığında çıkabilecek sorunlardan da etkilenmemektedir. Öte yandan, kısa tepki süresine sahiptir [22]. Parmakla çok hafif bir dokunuş ile kolaylıkla çalıştırılabilirler [21].

Yüzey akustik dalgalı ekranlar, kenarlara yerleştirilen dönüştürücüler tarafından bir cam plaka yüzeyine uygulanan ultrasonik dalgalardaki bozulmanın algılanmasına dayanır [20]. İki interdijital transdüser (interdigital transducer, IDT) dizisinden iletilen ultrasonik dalgalar, ekran yüzeyinin dokunulan kısmında sönümlenir [19]. Sinyal dalgalarındaki bu kesinti, dönüştürücüler tarafından algılanarak yüzeydeki konumun yorumlanması için denetleyiciye gönderilir [23].

Yüzey akustik dalga teknolojisini kullanan dokunmatik ekranlar, temas eden makul büyüklükteki herhangi bir nesneyi tanıyabilmektedir. En büyük olumlu yönleri, ekran boyunca ek katmana ihtiyaç duymamaları ve bu sayede ekran görüntüsünün tamamının iletilmesine imkân vermeleridir [24]. Bunun bir diğer getirisi ise, keskin nesnelere karşı dirençli dokunmatik ekranlara kıyasla daha dayanıklı olmasıdır. Ayrıca birden fazla temas noktasını aynı anda algılayabilme özelliğine de sahiptir [19].

Yüzey akustik dalgalı dokunmatik ekran panelleri, gelişmiş olmalarına rağmen dış unsurlardan kolaylıkla etkilenebilmektedir. Ekran yüzeyinde biriken kirletici unsurlar, ultrasonik dalgalarda girişime neden olabilmektedir. Ayrıca ekranın kenarları boyunca yerleştirilen verici ve alıcı dönüştürücüler tamamen kapatılamadıkları için büyük miktarlarda sıvı, kir veya tozdan zarar görebilmektedir [19]. Öte yandan, yüzey akustik dalga tasarımı daha pahalı bir teknolojidir [24].

Kızılötesi dokunmatik ekranlar, zıt kenarları boyunca yerleştirilen bir dizi kızılötesi LED ve fotodedektör çifti içerir [19]. Işık yayan diyotlar, ışın modelindeki bir kesintiyi algılamak için ışık dedektörleriyle eşleştirilir. Kullanıcı ekrana dokunduğunda ışık hüzmeleri kesilir ve dokunmanın tam yeri algılanır [20].

Kızılötesi dokunmatik ekrandaki teknoloji, parmak, eldiven veya kalem gibi birçok farklı nesnenin dokunuşunu algılayabilir. Kapasitif veya dirençli teknolojilerin aksine, cam üzerinde herhangi bir desen katmanına ihtiyaç duymaz. Bu durum, kızılötesi ekranların dayanıklılığını ve optik netliğini artırmaktadır. Kızılötesi teknolojisi, genellikle iletken nesne ile etkileşim gerektiren kapasitif teknolojinin uygulanamadığı durumlarda kullanılmaktadır [19].

### 2.1.2 Mikrofonlu Kulaklık

Kulaklık, bir elektriksel ses sinyalini akustik sese dönüştürmenin yoludur. Kulaklıklar, elektrodinamik hoparlörün dönüştürücü ilkesini küçük bir ölçekte kullanmaktadır. Sıradan bir hoparlöre göre daha büyük empedansa sahip olduklarından daha az güç ile çalışabilmektedirler [25].

Kulaklık, bir mıknatıs üzerine yerleştirilen bakır bobin ve bu bobine bağlı plastik zardan oluşur. Bobin üzerinden geçen akımın şiddetine göre oluşturduğu manyetik alan ile plastik zarı mıknatısa doğru iter ya da çeker [26]. Saniyenin çok küçük bir kısmında gerçekleşen bu hareket sayesinde plastik zarın havada oluşturduğu titreşim ses dalgalarını meydana getirmektedir.

Bir mikrofonun işlevi, havada yayılan ses sinyalini (basınç) karşılık gelecek bir elektrik sinyaline (gerilim) dönüştürmektir [25]. Bu çalışmada Şekil 2.2'de gösterilen Bluetooh bağlantı yöntemi ile çalışan kablosuz mikrofonlu kulaklık kullanılmıştır. Tek kart bilgisayara, Şekil 2.3'te gösterilen Bluetooth ile Wi-Fi (Wireless Fidelity, Kablosuz Bağlantı) bağlantı desteği kazandıran adaptör takılmıştır.

Şekil 2.2. Bluetooth destekli kablosuz mikrofonlu kulaklık.

Şekil 2.3. Bluetooth ve Wi-Fi adaptörü.

## 2.2 ÇEVRE HAKKINDA BİLGİ ALMA BİRİMİ

Kullanıcının hareket komutlarının sistem üzerinde oluşturduğu etkilerin tespit edilebilmesi ve beklenmedik durumlarda hızlı tepki verilebilmesi için bir geri besleme bileşenine ihtiyaç duyulmaktadır. Bu amaçla, ortamın derinlik haritasının çıkarılmasını sağlayan üç boyutlu derinlik algılayıcı kamera kullanılmıştır.

### 2.2.1 Derinlik Algılayıcı Kamera

Derinlik algılayıcı kameralar, normal kameralara eklenen derinlik sensörlerinden oluşmaktadır. Derinlik sensörleri, bir kızılötesi projektör ve algılayıcısından oluşmaktadır. Kızılötesi projektörün yansıttığı homojen olmayan desenler, algılayıcı tarafından okunmaktadır. Bu okunan desenlerin boyutları hesaplanarak mesafe bilgisi elde edilmektedir. Elde edilen mesafe bilgilerinin bir araya getirilmesiyle de derinlik haritası oluşturulmaktadır. Normal kamera görüntüsü ile derinlik haritasını bir araya getiren derinlik algılayıcı kameralar, görüş alanı içerisindeki nesnelerin ayırt edilerek işlenebilmesine imkân tanımaktadır. Derinlik algılayıcı kameraların olumsuz yönü ise, kızılötesi algılayıcılarının çok yoğun gün ışığı altında ayırt ediciliğini kaybetmesi ve benzer iki derinlik algılayıcı kameranın birbiri üzerinde gürültü oluşturmasıdır.

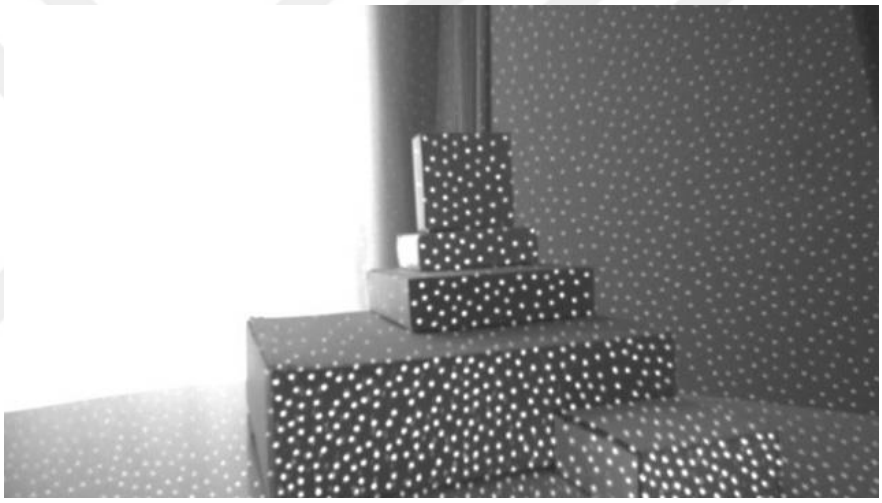
Bu tez çalışmasında derinlik algılayıcı kamera olarak, Şekil 2.4'te gösterilen Intel RealSense D435 kullanılmıştır. İlgili kameranın teknik özellikleri Çizelge 2.1'de listelenmiştir [27]. Bu kameradan elde edilen renkli, kızılötesi ve derinlik yayınından örnekler ise sırasıyla Şekil 2.5, Şekil 2.6 ve Şekil 2.7 üzerinde gösterilmiştir.

Çizelge 2.1. Intel RealSense D435 teknik özellikleri.

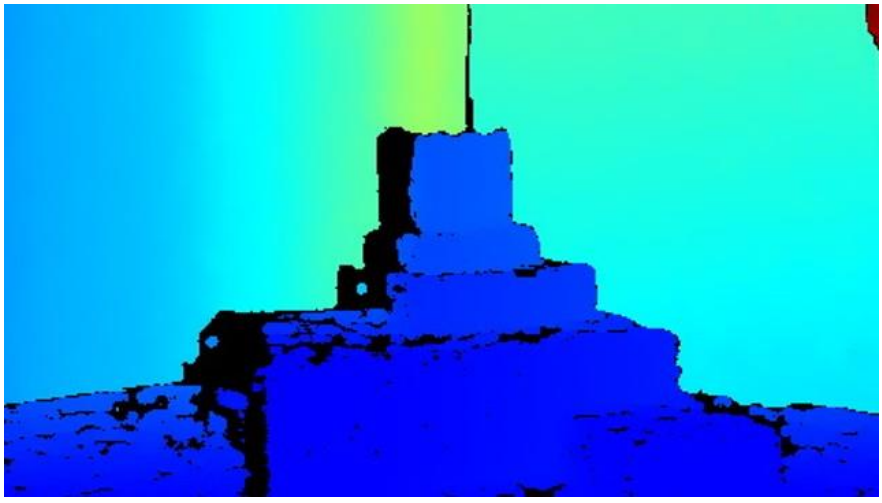
| | |
|---|---|
| **Kullanım Ortamı** | İç/Dış Mekân |
| **Görüntü Sensörü Teknolojisi** | Global Shutter, 3 µm × 3 µm piksel boyutu |
| **Maksimum Mesafe** | Yaklaşık 10 metre.<br>Doğruluk, kalibrasyona, sahneye ve aydınlatma koşullarına bağlı olarak değişir. |
| **Derinlik Teknolojisi** | Active IR Stereo |
| **Minimum Derinlik Mesafesi (Min-Z)** | 0.105 m |
| **Derinlik Görüş Alanı (FOV)** | 86° × 57° (±3°) |
| **Derinlik Çıktı Çözünürlüğü** | 1280 × 720'ye kadar |
| **Derinlik Çerçeve Hızı** | 90 fps'ye kadar |
| **RGB Sensör Çözünürlüğü** | 1920 × 1080 |
| **RGB Çerçeve Hızı** | 30 fps |
| **RGB Sensör FOV (H × V × D)** | 69.4° × 42.5° × 77° (±3°) |
| **Kamera Modülü** | Intel RealSense Modülü D430 + RGB Kamera |
| **Görüntü İşlemcisi** | Intel RealSense Vision Processor D4 |
| **Biçim Faktörü** | Kamera Çevre Birimi |
| **Uzunluk × Derinlik × Yükseklik** | 90 mm × 25 mm × 25 mm |
| **Bağlantı** | USB-C 3.1 Gen 1 |
| **Montaj Mekanizması** | –Bir 1/4-20 UNC vida dişi montaj noktası.<br>–İki M3 dişli montaj noktası. |

Şekil 2.4. Akıllı tekerlekli sandalye sisteminin sol ön tarafına takılan derinlik algılayıcı kamera.

Şekil 2.5 Intel RealSense D435 renkli yayın akışından örnek bir görüntü.



Şekil 2.6. Intel RealSense D435 kızılötesi yayın akışından örnek bir görüntü.



Şekil 2.7. Intel RealSense D435 derinlik yayın akışından örnek bir görüntü.

## 2.3 KARAR VERME BİRİMİ

Bu çalışma kapsamında, sistemin çalışmasını sağlayan algoritmaların işletilebilmesi, kullanıcı komutlarının alınabilmesi, görsel arayüzün sunulabilmesi, sensörlerden gelen ham verinin işlenebilmesi, sinyalizasyonun yapılabilmesi ve motorların sürülebilmesi için bir bilgisayar ile buna bağlı mikrodenetleyiciler sisteme dâhil edilmiştir. Bilgisayar olarak, gerekli tüm temel bilgisayar bileşenlerini içeren tek kart bilgisayar kullanılmıştır. Mikrodenetleyici olarak ise, hızlı prototip geliştirme imkânı sunan tek kart mikrodenetleyiciler tercih edilmiştir.

### 2.3.1 Tek Kart Bilgisayar

Tek kart bilgisayar, bir bilgisayar anakartının küçültülerek üzerine gömülü bir işlemci eklenmesinden oluşmaktadır [28]. Tek kart bilgisayarlar, üzerinde bir işletim sistemi çalıştırarak çoklu görev işlemlerini gerçekleştirebilmektedir [29]. Günümüzde farklı markalara ait birden çok tek kart bilgisayar bulunmaktadır. Bu tez çalışmasında, Şekil 2.8'de gösterilen UPBoard tek kart bilgisayarı tercih edilmiştir. UPBoard teknik özellikleri Çizelge 2.2'de listelenmiştir [30].

Şekil 2.8. UPBoard tek kart bilgisayarı.

Çizelge 2.2. UPBoard teknik özellikleri.

| | |
|---|---|
| **Yongada Sistem** | Intel Atom x5-Z8350 Processor<br>(2M Önbellek, 1.44GHz, 1.92GHz'e kadar)<br>CPU, 64 bit mimari, dört çekirdekli |
| **Bellek** | 1GB / 2GB / 4GB DDR3L-1600 |
| **Grafik** | Intel HD 400 Graphics, 12 EU GEN 8,<br>500MHz'e kadar, DX*11.1/12, Open GL*4.2,<br>Open CL*1.2, OGL ES3.0, H.264,<br>HEVC (decode), VP8 destekli |
| **Depolama Kapasitesi** | 16GB / 32GB / 64GB eMMC |
| **Video ve Ses** | HDMI 1.4b, I2S ses bağlantı noktası |
| **Ekran Arayüzü** | DSI / eDP |
| **Camera Arayüzü** | CSI (4 Megapiksel) |
| **Ethernet** | 1× Gbit Ethernet (tam hız) RJ-45 |
| **USB 2.0** | 4× USB 2.0,<br>2× USB 2.0 pin başlığı (toplamda 10 pin) |
| **USB 3.0** | 1× USB 3.0 OTG |
| **Gerçek Zamanlı Saat** | Evet |
| **Genişletme** | 40 pin genel amaçlı bus, Altera Max V destekli,<br>A-D Dönüştürücü, 8-bit 188ksos |
| **Güç** | 5 V DA, 3 A 5.5/2.1 mm giriş |
| **Uyumlu İşletim Sistemleri** | Microsoft Windows 10, Windows IoT Core,<br>Linux (ubilinux, Ubuntu, Yocto),<br>Android Lollipop, Brillo |
| **Çalışma Nemi** | %10~%80 RH yoğuşmasız |
| **Çalışma Sıcaklığı** | 32-140°F / 0~60°C |
| **Boyutlar** | 3.37" × 2.22" / 85.60 mm × 56.5 mm |
| **Sertifikalar** | CE/FCC Class A, RoHS belgesi,<br>Microsoft Azure sertifikası |

### 2.3.2 Tek Kart Mikrodenetleyici

Mikrodenetleyici, bir bilgisayarı oluşturan en temel bileşenlerin küçük boyutlu tek bir tümleşik devre içerisinde üretilmiş hâlidir. Bir mikroişlemciye, verileri tutmak için RAM'e (Random Access Memory, Rastgele Erişim Belleği), programları tutmak için EPROM'a (Erasable Programmable Read-Only Memory, Silinebilir Programlanabilir Salt okunur Bellek) ya da flash belleğe ve haricî cihazlarla arayüz oluşturmasına izin

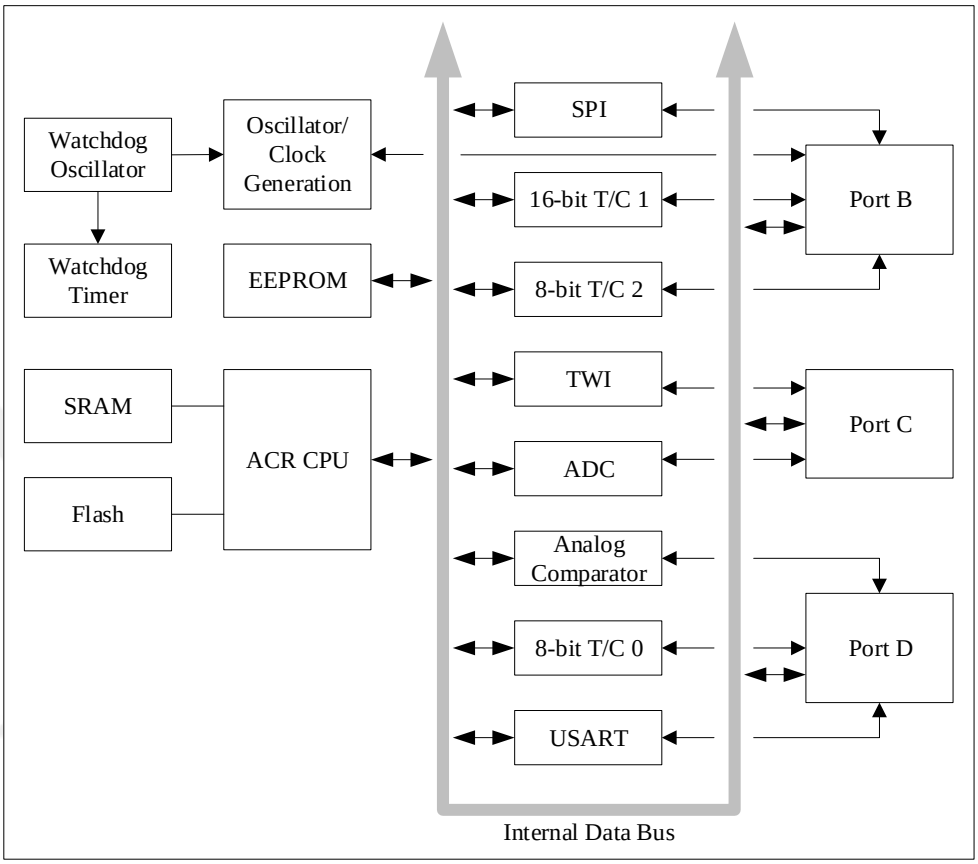
veren giriş/çıkış bağlantı noktalarına sahiptir [31]. Mikrodenetleyiciler, küçük ve kullanışlıdır [32]. Toplu üretimleri ucuz olup gömülü uygulamalar için özel olarak tasarlanmıştır. Günümüzdeki birçok elektronik eşyada kullanılmaktadır [33].

Bir mikrodenetleyicinin, çalışması için sabit bir elektrik kaynağına ve programlanması için bilgisayarla iletişim kurmasını sağlayan bazı destekleyici elektronik elemanlara ihtiyacı vardır [31]. Bu elemanların mikrodenetleyiciye eklenebilmesi için özelleştirilmiş bir kart tasarlanmalı ya da bir geliştirme kartı temin edilmelidir. Bir mikrodenetleyici geliştirme kartı, dâhili mikrodenetleyiciye ve bu mikrodenetleyici için gereken destekleyici çevre birimlerine sahip bir baskı devre kartıdır. Bu kartlar, mikrodenetleyici ve destek elektroniğinin yanı sıra LED'ler, anahtarlar ve erkek-dişi bağlantıları gibi farklı ek özellikler de içerebilmektedir [28]. Arduino, yaygın mikrodenetleyici geliştirme kartlarından biridir ve çoğunlukla 8 bitlik Atmel AVR mikrodenetleyicisi barındırır [29].

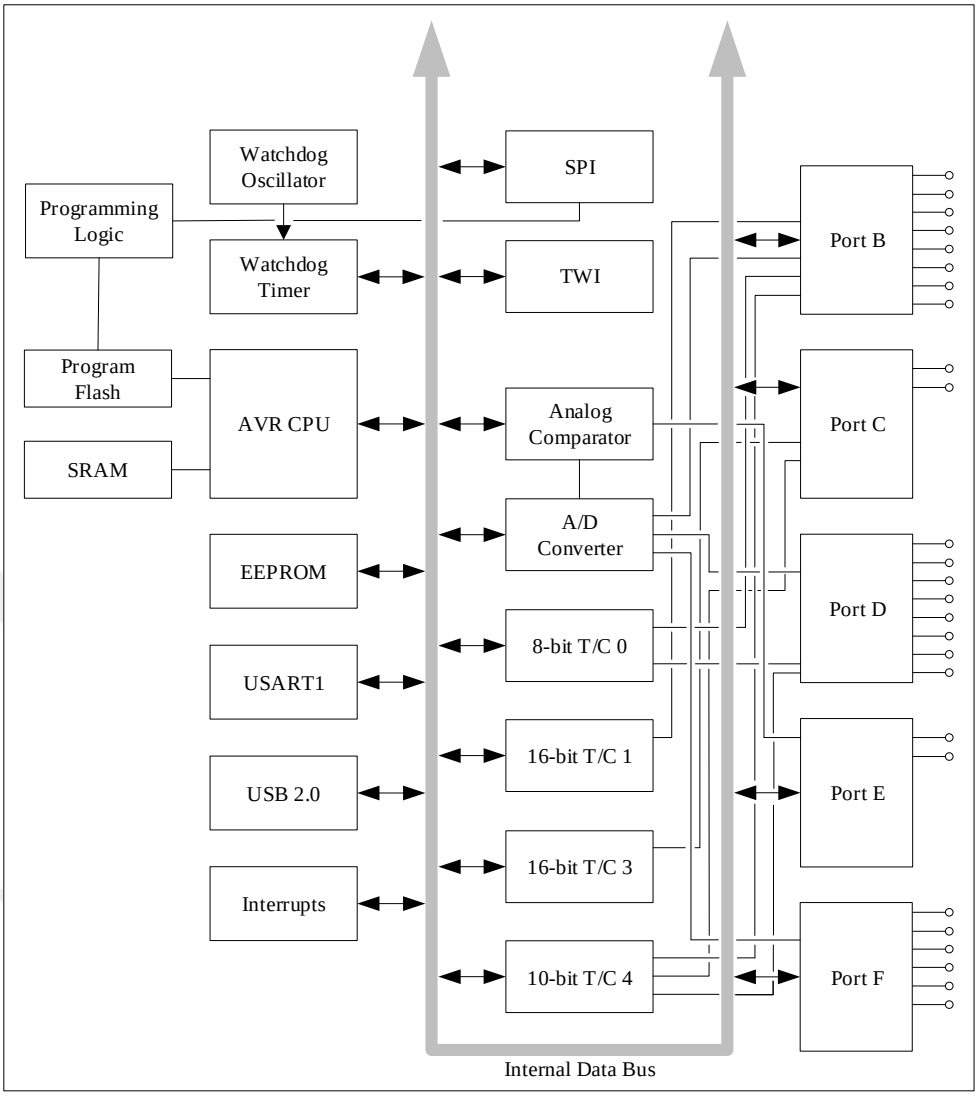
Atmel (Advanced Technology for Memory and Logic), 1984 yılında kurulmuş bir Amerikan yarı iletken şirketidir. Şirket, Alf-Egil Bogen ve Vegard Wollan tarafından tasarlanan RISC tipi iç mimariye sahip AVR mikrodenetleyicisini 1996'da piyasaya sürmüştür. AVR, program depolaması için yonga üzerinde flaş belleği kullanan ilk mikrodenetleyici ailelerinden biridir [34]. AVR mikrodenetleyicilerinin çoğu, flash bellekteki değişiklikler esnasında kalıcı olması gereken işletim parametreleri gibi verileri depolamak için az miktarda EEPROM'a (Electrically Erasable Programmable Read-Only Memory, Elektriksel Silinebilir Programlanabilir Salt Okunur Bellek) sahiptir [35].

Aynı merkezî işlem birimi (Central Processing Unit, CPU) komutları kullanan AVR çeşitleri arasındaki temel farklar, yerleşik flash bellek miktarı, en yüksek saat hızı, yongadaki giriş/çıkış pinlerinin sayısı ve dâhili çevre birimi işlevlerine dayanmaktadır. Bu tez çalışmasında, tek kart bilgisayardan komut alınması için Atmel ATmega328 mikrodenetleyicileri ve tek kart bilgisayara geri bildirim iletilmesi için Atmel ATmega32U4 mikrodenetleyicisi kullanılmıştır. Şekil 2.9'da dâhilî yapısı gösterilen ATmega328, 32 KB flaş bellek, 2 KB SRAM ve 1 KB EEPROM belleğe sahiptir [34]. Şekil 2.10'da dâhilî yapısı gösterilen ATmega32U4 ise, genel olarak ATmega328'e

benzer olup USB iletişiminin yürütülmesi için ek parça ihtiyacını ortadan kaldıran yerleşik bir USB arayüzüne sahiptir [35].




Şekil 2.9. ATmega328 mikrodenetleyicisinin dâhilî blok diyagramı.



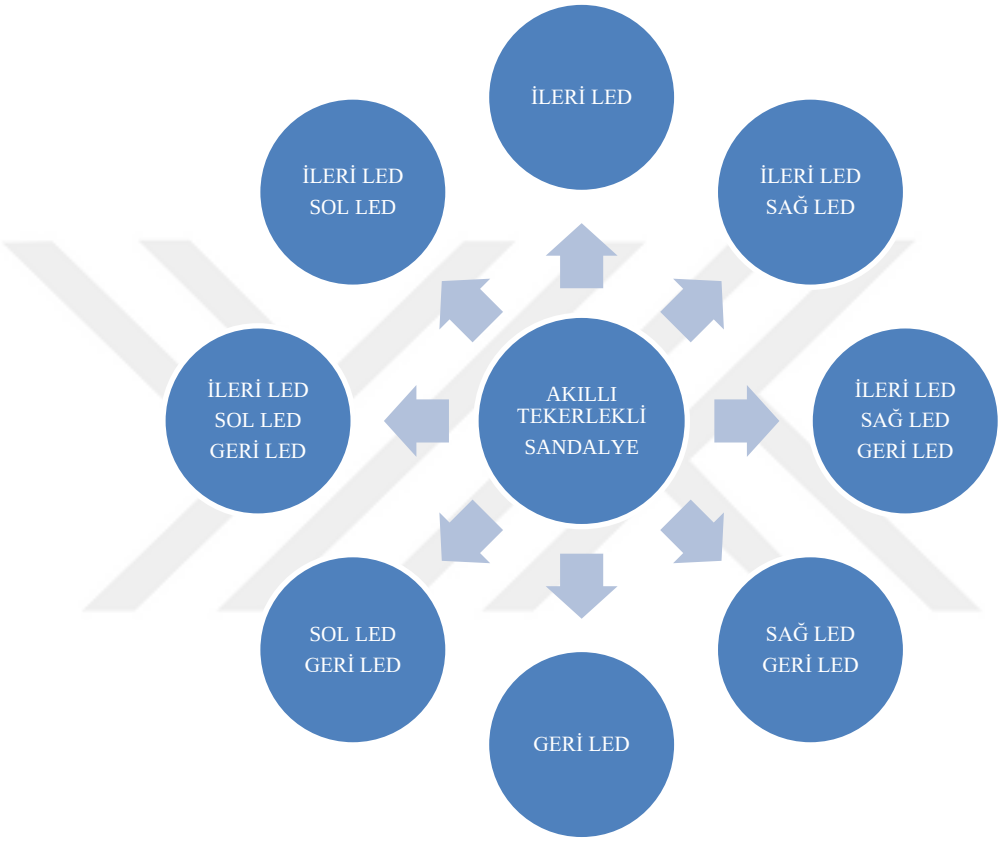

Şekil 2.10. ATmega32U4 mikrodenetleyicisinin dâhilî blok diyagramı.

## 2.4 SİNYALİZASYON BİRİMİ

Sinyalizasyon sistemi, akıllı tekerlekli sandalyenin hareketi hakkında etrafındakilerin uyarılmasını sağlamaktadır. Bu çalışmada, görsel sinyalizasyon için LED aydınlatmalar ve sesli sinyalizasyon için de vızlayıcı (buzzer) kullanılmıştır.

### 2.4.1 LED Aydınlatmalar

Akıllı tekerlekli sandalye üzerine, kara taşıtlarında kullanılan ışık sistemlerindeki gibi LED lambalar eklenmiştir. Bu LED aydınlatma sistemi, akıllı tekerlekli sandalyenin hareket yönüne göre Şekil 2.11'deki gibi çalışmaktadır.




Şekil 2.11. Akıllı tekerlekli sandalye sisteminin hareket yönleriyle ilişkilendirilen LED aydınlatmalar.

Akıllı tekerlekli sandalye sisteminin önünün aydınlatılması, arkadan görünürlüğünün arttırılması, acil durma durumlarında ve dönüşlerinde uyarı sinyallerinin verilebilmesi için ön ve arka tarafa LED aydınlatmalar eklenmiştir. Akıllı tekerlekli sandalye sisteminin ön ve arkasına takılan LED lambalar Şekil 2.12'de gösterilmiştir.

(a) (b)

Şekil 2.12. Hazırlanan akıllı tekerlekli sandalye sisteminin önden (a) ve arkadan (b) görünümü.

### 2.4.2 Vızlayıcı

Vızlayıcı (buzzer), içerisinde bulunan armatürün titreşmesiyle gürültülü bir ses üreten sinyal cihazıdır [36]. Akıllı tekerlekli sandalye sisteminin geri hareketinde aralıklı ses çıkarması ve önüne engeller çıkması durumunda sesli ikaz vermesi için Şekil 2.13'te gösterilen bir adet vızlayıcı tek kart mikrodenetleyiciye takılmıştır. Acil durma durumlarında ise vızlayıcı düzenli aralıklarda ses çıkararak sistemin duraklatıldığını bildirmektedir.

Şekil 2.13. Vızlayıcı.

## 2.5 HAREKET KONTROL BİRİMİ

Akıllı tekerlekli sandalye sisteminin hareketi, sahip olduğu iki adet DA motor tarafından sağlanmaktadır. Bu motorların sürülebilmesi ve hassas devir ayarının yapılabilmesi için dijital kontrole imkân tanıyan haricî çift DA motor sürücü kartı ve tekerleklerin devir sayısını ölçen hız sensörü kullanılmıştır.

### 2.5.1 DA Motor Sürücüsü

Bu tez çalışmasında, çalışma gerilimi aralığı 3-36V, nominal çalışma akımı 15A, anlık tepe akımı 30A olan ve aynı anda iki adet DA motoru sürebilen bir sürücü kartı kullanılmıştır. Bu sürücü kartı, Şekil 2.14'te gösterildiği gibi sekiz adet IRF3205 MOSFET içermektedir. Çift DA motor sürücü kartı, motorları ileri-geri hareket ettirerek hızını ayarlayabilmek için birbirinden yalıtılmış iki adet H-köprüsü barındırmaktadır. Bu H-köprülerinin üst ve alt anahtarları da birbirinden elektriksel olarak optokuplör (optocoupler) yardımıyla izole edilmiştir. Besleme, yön ve DGA (Darbe Genişlik Ayarlı) girişleri olmak üzere toplamda 6 adet girişe sahiptir. Yön kontrol işlemleri kendi içerisinde tasarlanmış lojik kapı entegreleri vasıtası ile yapılmaktadır. %0-99 görev oranlarında motorların terminal uçlarını besleyebilmektedir. İyi bir soğutma işlemi uygulandığında üzerinden akan akıma karşı dayanıklı olması, ısınma sorununun fazla olmaması ve maliyetinin uygun olması nedeniyle bu çift DA motor sürücü kartı tercih edilmiştir.

Şekil 2.14. Çift DA Motor Sürücü Kartı.

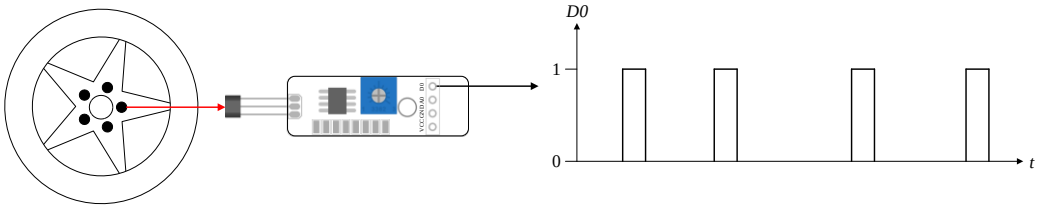
Akülü tekerlekli sandalye motorları, güvenlik önlemi olarak içerisine yerleştirilen manyetik fren sisteminden dolayı doğrudan motor sürücüsü ile hareket ettirilemez. Motorların hareket edebilmesi için öncelikle dâhilî fren sisteminin devre dışı bırakılması ve motor milindeki balata baskısının kaldırılması gerekir. Bunun için motorun fren sisteminde bulunan bobine gerilim uygulanmalı ve oluşturulan yüksek manyetik alan etkisiyle balataya baskı yapan demir plaka geriye çekilmelidir. Dolayısıyla, çift DA motor sürücü kartı ile motor hareket ettirilirken aynı zamanda röleler yardımıyla da fren sistemlerine aküden gerilim uygulanmaktadır.

### 2.5.2 Hız Sensörü

Bir tekerlekli sandalyenin konum bilgisi elde edilirken, uydu bağlantısı gerektiren küresel konumlandırma sistemi (Global Positioning System, GPS) kullanılabilmektedir. Bu tez çalışmasında ise farklı olarak tekerlek odometrisi tekniği kullanılmış ve itici tekerlere takılan hız sensörlerinden alınan veriler değerlendirilmiştir.

Odometri, bilinen başlangıç konumuna göre yapılan değişikliklerin dolaylı yollardan ölçülmesiyle mevcut konumun belirlenmesidir [37]. Odometri yöntemlerinden tekerlek odometrisinde ise, her bir tekerleğin yaptığı devir sayısı ile konum değişiklikleri ilişkilendirilir. Tekerlek odometrisi, her tekerlek için bağımsız olarak farklı hız ve yönlerde çalıştırılabilmektedir. Bu yöntemin kullanıldığı yerlere örnek olarak NASA'nın (U.S.A. National Aeronautics and Space Administration, A.B.D. Ulusal Havacılık ve Uzay Dairesi) MER (Mars Exploration Rover, Mars Keşif Aracı) sistemleri gösterilebilir [38].

Hall etkili manyetik sensör, üzerinden akım geçen iletken bir levhaya yaklaştırılan bir mıknatısın gerilim farkı ölçülebilen pozitif ve negatif kutuplar oluşturmasına dayanır. Bu tez çalışmasında, akıllı tekerlekli sandalye sistemi için iki adet A3144 Hall etkili manyetik sensör, motora bağlı tekerlere takılmış ve Şekil 2.15'te gösterildiği gibi teker dönüşünü ölçen birer hız sensörü olarak kullanılmıştır.




Şekil 2.15. Hall etkili manyetik sensör ve tekere takılan mıknatıslara verdiği tepki.

Tekerleğin yarıçapı $r$ olarak kabul edilirse, bir tam turda aldığı yol dairenin çevresinden $2\pi r$ olarak hesaplanır. Sensör tarafından mikrodenetleyiciye gönderilen darbe sinyalleri arasındaki tekerlekli sandalyenin katettiği mesafe ölçümü için çözünürlük, tekerlek jantına eşit aralıklarda yerleştirilen beş mıknatıs ile $2\pi r/5$ olarak belirlenmiştir.

Tekerlek odometrisi gerçek zamanlı geri besleme sunmaktadır. Buna rağmen birtakım olumsuz yönlere sahiptir. Yer değişimi ile tekerlek dönüşü arasındaki ilişki her zaman doğrusal değildir. Tekerlek kayması nedeniyle karmaşık engebeli arazilerde ve kaygan yüzeylerde hatalar olabilmektedir. Ölçümlerdeki hatanın zamanla birikmesi sonucunda ise ciddi bir konum kayması ile karşılaşılabilir. Bu nedenle ikincil bir yöntemle desteklenmesine ihtiyaç vardır. Bu tez çalışmasında, derinlik algılayıcı kamera ile alınan verilerden çevreye özgü bilgiler çıkarılarak konum ve yön tahmini iyileştirilmektedir. Doğrudan veya dolaylı teknikler yardımıyla daha kesin konum tahminlerinin elde edilmesi sağlanmaktadır [39].

# BÖLÜM 3

# SİMÜLASYON ORTAMININ HAZIRLANMASI

Araştırma alanlarının birçoğunda, gerçek uygulamadan önce teorilerin, fikirlerin ve tasarımların test edilmesi için simülasyon yazılımları kullanılmaktadır. Özellikle robotik alanında, gelişmiş donanımların pahalı olmasından dolayı simülasyon yazılımları ile denemelerin yapılması büyük önem taşımaktadır. Piyasada, farklı özelliklere sahip birçok 3B (Üç Boyutlu) robotik simülasyon yazılımı bulunmaktadır [40]. Bu simülasyon yazılımlarından açık kaynak erişimi sunanlara Webots [41], Gazebo [42], USARSim [43] ve CoppeliaSim [44] örnek verilebilir [45].

Bu tez çalışmasında, Coppelia Robotics tarafından geliştirilen ve V-REP (Virtual Robot Experimentation Platform, Sanal Robot Deney Platformu) olarak bilinen yazılımın devam sürümü olan CoppeliaSim yazılımı kullanılmıştır. Bu yazılım, ticari sürümünün tüm özelliklerini sunabilen eğitim amaçlı ücretsiz tam bir lisansa sahip olduğu için tercih edilmiştir [45]. Ayrıca CoppeliaSim yazılımı, sunduğu çapraz platform desteği sayesinde farklı işletim sistemlerinde de çalıştırılabilmektedir.

## 3.1 SİMÜLASYON ORTAMINA GENEL BAKIŞ

CoppeliaSim yazılımı, robot sistemlerinin ve algoritmaların prototiplenmesi, geliştirilmesi ve doğrulanması için kullanılan genel amaçlı, çok yönlü ve ölçeklenebilir bir robot simülasyon ortamıdır [46]. Bu yazılım, güvenlik, uzaktan izleme, eğitim-öğretim, donanım kontrolü ve fabrika otomasyonu gibi simülasyon durumları için hızlı algoritma geliştirme, sistem doğrulama, prototip oluşturma ve dağıtma dâhil pek çok uygulamada kullanılabilmektedir [47].

Öngörülemeyen özellikler barındıran karmaşık sistemleri işleyebilmesi için CoppeliaSim yazılımı, çeşitli kinematik, fizik ve grafik kitaplıklarından oluşturulmuş

çok sayıda araç ve fonksiyona sahiptir. Yol planlama, zamanı yolla ilişkilendirme ve çarpışmayı tespit etme amacıyla kullanıma hazır birkaç algoritma içermektedir [40]. Simülasyonun denetimi için gereken unsurlar yazılımın ana döngüsüne bağlı olarak yürütülmektedir. Özellikle gerçek zamanlı kontrol için gereken temel unsurlar, her simülasyon geçişinde ana simülasyon döngüsü ile eşzamanlı çalıştırılmaktadır [47].

CoppeliaSim yazılımındaki bir simülasyon, tek dosya üzerinden çalıştırılıp saklanabilmekte, kolayca kodlanarak güncellenebilmekte, modelleme içeriği zenginleştirilebilmekte ve farklı ortamlar için yeniden ölçeklendirilebilmektedir. Bu özellikleri sayesinde, CoppeliaSim yazılımı esnek bir denetleyici yaklaşımı sunmaktadır [47].

CoppeliaSim yazılımı, öğelerinin birbirleriyle olan etkileşimlerini belirleyebilmek için gömülü komut dosyalarının çalıştırılabildiği çok yönlü bir denetim mimarisine sahip olarak tasarlanmıştır. Bu yazılımda, farklı nesnelerin yüklenebildiği özelleştirilebilir bir sahne üzerinden tüm simülasyon içeriği modellenebilmektedir. Sahnedeki tüm nesneler, gömülü komut dosyaları kullanılarak programlanabilir [48]. Lua programlama dilinde yazılan gömülü komut dosyaları, ilişkilendirildiği nesneyle birlikte bir simülasyon modeli olarak saklanabilmektedir. Simülasyon modelleri, farklı sahnelerde yeniden kullanılmak üzere taşınabilmekte ve ölçeklendirilebilmektedir. Modellerin işlevleri gerektiğinde etkinleştirilebilmekte ya da devre dışı bırakılabilmektedir. İsteğe göre etkinleştirilen kısmen bağımsız işlevlerin uyarlanmasıyla, aynı anda birbirleriyle iş birliği yapacak şekilde kullanılabilmektedir. Modellerin işlevselliği, belirli sahne nesneleri veya hesaplama modülleri ile ayarlanabilmektedir [47]. Ayrıca CoppeliaSim yazılımı, modüler yapısına uygun olarak grafiksel kullanıcı arayüzü üzerinden bir tümleşik geliştirme ortamı (Integrated Development Environment, IDE) da sunmaktadır [49].

CoppeliaSim yazılımı, simülasyon modellerini hem daha kolay erişilebilir hâle getirmekte hem de sahneye yerleştirme karmaşıklığını azaltmaktadır [50]. Yazılım, simülasyon çalışırken etkileşim kurulabilmesi için kullanılabilecek çok sayıda örneğe, robot modeline, sensöre ve devindirici düzeneklere (aktüatörlere) sahiptir. Sahneye sürüklenip kolaylıkla eklenerek tamamen özelleştirilebilen bu modeller, yazılımda

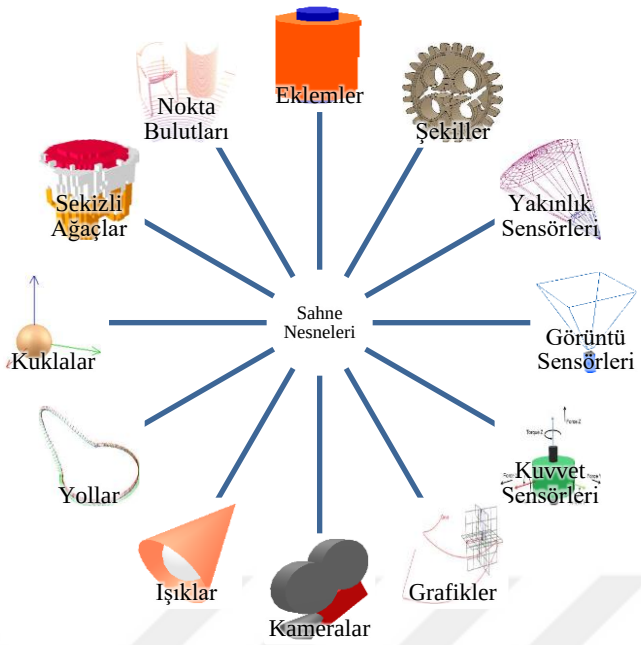
hâlihazırda yüklü bulunmaktadır. Buna ek olarak, yerleşik CAD (Computer-Aided Design, Bilgisayar Destekli Tasarım) modelleri arasında Microsoft Kinect gibi çeşitli ticari sensörler de vardır. Gerek duyulması hâlinde, yerleşik modellere benzer şekilde görüntü ve yakınlık sensörlerinin farklı kombinasyonları kullanılarak yeni simülasyon modelleri oluşturulabilmektedir [51]. Böylece, simülasyon sahnelerini özelleştirmek için tasarlanan yeni modeller CoppeliaSim yazılımına eklenebilmektedir.

Bunların yanı sıra, CoppeliaSim yazılımı çok sayıda programlama yaklaşımı sunarak üretkenliği artırmaktadır [47]. Simülasyon sahnesindeki her bir nesne ya da model, gömülü komut dosyası, uzantı, eklenti, uzak API (Application Programming Interface, Uygulama Programlama Arayüzü) istemcisi, ROS (Robot Operating System, Robot İşletim Sistemi) [52] düğümü, BlueZero düğümü veya özel bir çözüm aracılığıyla kontrol edilebilmektedir [51]. Uzak API eklentisi aracılığıyla, Lua programlama dilinden farklı programlama dilleri (C/C++, Python, Java, MATLAB, Octave ve Urbi) de kullanılabilmektedir. Bu sayede, kapsamlı bir programlama arayüzü sunmaktadır.

Bir robot simülasyonu yapılmak istendiğinde, bağlantılar, eklemler ve ters kinematik gibi hesaplama modülleri için kurulum yapılması gerekmektedir. CoppeliaSim yazılımının temel öğeleri, sahne nesneleri, hesaplama modülleri ve kontrol mekanizmaları olarak üç ana grupta incelenebilir. CoppeliaSim yazılımının ters kinematiği dört farklı dinamik motorunu desteklemektedir. Bunlar her biri farklı avantajlara sahip, Bullet Physics Library, Open Dynamics Engine, Vortex Studio ve Newton Dynamics simülasyon motorlarıdır. Simülasyon çalıştığı sırada, simülasyon sahnesi gerçek 3B dünya gibi davranmakta ve simülasyon modellerinin davranışına göre gerçek zamanlı geri bildirim vermektedir [51].

## 3.2 SAHNE NESNELERİ

CoppeliaSim yazılımında bir simülasyon sahnesi ya da simülasyon modeli, ağaç benzeri hiyerarşide birleştirilmiş birkaç sahne nesnesini (temel nesneyi) içermektedir. Bu yazılımda desteklenen sahne nesneleri Şekil 3.1'de gösterildiği gibidir [47].




Şekil 3.1. CoppeliaSim yazılımının sahne nesneleri.

### 3.2.1 Eklemler (Joints)

Eklemler, bir ile üç arası serbestlik derecesiyle (Degree of Freedom, DoF) iki veya daha fazla sahne nesnesini birbirine prizmatik, döner, vidalı ya da küresel olarak bağlayan nesnelerdir. Mekanizmaların oluşturulmasında ve nesnelerin hareket ettirilmesinde kullanılırlar. Kuvvet, tork ve ters kinematik hesaplamalarında görev alırlar [47]. Ayrıca eklemler, yalnızca bir eksen boyunca iletilen kuvveti veya bir eksen çevresinde oluşan torku ölçebilmektedir [53].

### 3.2.2 Şekiller (Shapes)

Şekiller, simülasyonda sert gövde görselleştirmesi için kullanılan üçgen ağlardır. Hızlı dinamik çarpışma tepkisini hesaplamak için birden fazla şekil gruplanarak uygun hâle getirilebilmektedir. Simülasyondaki diğer sahne nesneleri ve tüm hesaplama modülleri, şekillere göre hesaplanarak yürütülür [47]. CoppeliaSim yazılımı ile şekiller, içe veya dışa aktarılabilmekte ve düzenlenebilmektedir.

Şekiller, rastgele, bombeli (dışbükey), düz ve yükseklik alanı olmak üzere dört farklı gruba ayrılır. Düzensiz bir ağı temsil eden rastgele şekiller, dinamik çarpışma

tepkisinin hesaplanması için uygun değildir. Simülasyonu yavaşlattığı ve kararsız hâle getirdiği için rastgele şekillerin dinamik çarpışmada kullanılması önerilmez. Dışbükey ağı temsil eden bombeli şekiller ise, dinamik çarpışma tepkisi hesaplaması için iyileştirilmiştir. Öte yandan küp, silindir veya küreyi temsil eden düz şekiller, dinamik çarpışma tepkisi hesaplaması için en uygun olanıdır. Simülasyonu hızlı ve kararlı olduğundan dinamik çarpışma hesabı için düz şekiller önerilmektedir. Yükseklik alanı şekilleri, yalnızca yüksekliğin değişebildiği ızgara olarak bir yüzeyi temsil etmektedir. Bu şekiller dinamik çarpışma tepkisi için iyileştirilmiş olup birer düz şekil olarak da görülebilmektedir [53].

### 3.2.3 Yakınlık Sensörleri (Proximity Sensors)

Yakınlık sensörleri, yapılandırılabilir algılama hacmi içerisindeki bir şekil parçası için en kısa mesafeyi hesaplayarak kesin değer sunar [54]. Bu sensörler kullanıldığında, simülasyonu daha gerçekçi kılan devamlı bir ölçüm süreci elde edilebilir [47]. Piramit, silindir, daire, koni ve ışın tipi yakınlık sensörlerini destekleyen CoppeliaSim yazılımı ile ultrasonik ve kızılötesi gibi birçok farklı türde sensörler modellenebilmektedir [53].

### 3.2.4 Görüntü Sensörleri (Vision Sensors)

Görüntü sensörü, ışıklara, renklere ve görüntülere tepki veren kamera tipi bir sensördür. Kendi görüş alanındaki nesneleri işleyerek çekimin belirtilen eşikten fazla ya da az olması durumunda algılamayı tetikler [53]. Bu sensörler, bir simülasyon sahnesinden renkler, nesne boyutları ve derinlik haritaları gibi karmaşık görüntü bilgilerinin çıkarılmasında kullanılır. CoppeliaSim yazılımında yerleşik bir filtreleme ve görüntü işleme fonksiyonu bulunmaktadır. Görüntü sensörleri, ham görüntü alımı için OpenGL (Open Graphics Library, Açık Grafik Kütüphanesi) donanım hızlandırmasından yararlanır [47]. Kullanılan grafik kartına veya simülasyon sahnesindeki nesnelerin karmaşıklığına bağlı olarak, görüntü sensörleri yakınlık sensörlerine göre biraz daha yavaş çalışabilmektedir.

### 3.2.5 Kuvvet Sensörleri (Force Sensors)

Kuvvet sensörü, kendisine uygulanan kuvvetleri ve torkları ölçebilen bir nesne olup şekiller arasındaki temel katı bağlantıları temsil eder. Bu sert bağlantılar, uygulanan kuvvetleri ve torkları kaydederek belirli bir eşik değeri aşıldığında koşullara bağlı olarak kırılırlar [47]. Kuvvet sensörleri, simülasyon sırasında dinamik olarak etkinleştirilir. Bu sensörler, eklemlerden farklı olarak her bir eksen boyunca iletilen kuvveti ve her bir eksen etrafında oluşan torku ölçebilmektedir [53].

### 3.2.6 Grafikler (Graphs)

Grafikler, simülasyon verilerini kaydetmek, görselleştirmek veya dışa aktarmak için kullanılan nesnelerdir. Belirli nesnelere uygulanan çok çeşitli veri akışları (sıralı veri değeri listesini) grafikler ile kayıt edilebilmektedir [47]. Kayda alınan veri akışları, zaman grafiği (zamana karşı veri akışı A), X/Y grafiği (veri akışı A karşı veri akışı B) ya da 3B eğri grafiği (zamana karşı veri akışları A, B ve C) olarak görüntülenmesi için birleştirilirler [53].

### 3.2.7 Kameralar (Cameras)

Kamera, simülasyon sahnesinin çeşitli bakış açılarından görülmesini sağlayan bir nesnedir. Görüntü sensörleri ile kameraların karıştırılmaması gerekir. Kameranın belirli bir çözünürlüğü bulunmazken, görüntü sensörünün sabit bir çözünürlüğü vardır. Bir API aracılığıyla kameranın görüntü içeriğine erişilemezken, görüntü sensörünün görüntü içeriğine erişilebilir. Görüntü sensörleri yalnızca işlenebilir nesneleri görüntüleyebilirken kameralar tüm nesneleri daha hızlı bir şekilde görüntüleyebilmektedir. Ayrıca hareketli bir nesneyi takip etmesi için kameralar kullanılabilmektedir. Bu amaçla ayarlanan kameralar, konumlarını korurken ilgilendikleri nesneyi izleme alanı içerisinde tutmak için baktığı yönü otomatik olarak değiştirmektedir [53].

### 3.2.8 Işıklar (Lights)

Işıklar simülasyon sahnesini veya bağımsız sahne nesnelerini aydınlatmada kullanılır. Kameralar ve görme sensörleri, ışıklardan doğrudan etkilenir [47]. Sahnede herhangi bir ışık nesnesi olmadığında, tüm nesneler ortam ışığıyla aydınlatılarak gölgesiz renklerde gösterilmektedir. CoppeliaSim yazılımında, ışıldak (spotlight), çok yönlü ışık (omnidirectional light) ve yönlü ışık (directional light) olmak üzere üç farklı ışık türü vardır. Işıldaklar, sahneyi belirli bir yönde ayarlanabilen ışık konisi üzerinden aydınlatır. Çok yönlü ışıklar, sahneyi bir nokta ışık kaynağından her yöne doğru aydınlatmaktadır. Yönlü ışıklar ise konumundan bağımsız olarak sahnedeki tüm nesneleri belirlenen bir yönden aydınlatmaktadır [53].

### 3.2.9 Yollar (Paths)

Yol, birleştirilebilen takip etme, dönme veya duraklama gibi karmaşık hareketlerin tanımlandığı bir nesnedir. Bir robotun önceden tanımlanmış bir yörünge boyunca yönlendirilmesi veya bir konveyörün bant hareketlerinin sağlanması gibi çeşitli görevler için yollar kullanılabilir [47]. İstenilen etkinin elde edilebilmesi için yolların genellikle kuklalar ve diğer nesnelerle birlikte kullanılması gerekir [53].

### 3.2.10 Kuklalar (Dummies)

Birçok farklı uygulaması bulunan kukla, diğer sahne nesneleri veya hesaplama modülleriyle birlikte kullanılabilen çok amaçlı bir yönelimli noktadır. Genellikle kuklalar, yardımcı bir referans çerçevesi olarak da görülmektedir [53].

### 3.2.11 Sekizli Ağaçlar (OC Trees)

Hiyerarşik veri yapılarında, dörtlü ağaç ve sekizli ağaç yaygın olarak kullanılmaktadır. Dörtlü ağaçlar iki boyutlu uygulamalarda kullanılırken sekizli ağaçlar ise üç boyutlu uygulamalarda kullanılır. Bu ağaç yapıları, büyük ölçüde aynı veya benzer değerlere sahip verilerin bir araya getirilerek depolamadan tasarruf edilmesi için geliştirilmiştir.

Görüntü işlemede dörtlü ağaç ve sekizli ağaç yapıları sayesinde işlemlerin çoğu ağaç geçişleri uygulanarak gerçekleştirilebilir [55].

Sekizli ağaç (octal-tree / oct-tree / octtree / octree / oc tree), dörtlü ağacın üç boyutlu bir genellemesidir [56,57]. Konumları ile boyutları sınırlı kübik bir mozaik kullanılarak ayrıştırılan nesne hacimleri, değişken kübik bloklar kümesinin birleşimi olarak ifade edilir [58]. Bu ağaç veri yapısında her bir düğüm tam olarak sekiz dala sahiptir. Dolu yaprak düğümleri voksel olarak temsil edilir. Sekizli ağaçlar, şekiller veya nokta bulutlarının basitleştirilmesini sağlamaktadır. Bununla birlikte, simülasyonun çarpışma algılamalarında, en kısa mesafe hesaplamalarında ve yakınlık sensörlerince tespit edilmesinde kullanılabilmektedirler [53].

### 3.2.12 Nokta Bulutları (Point Clouds)

Nokta bulutu, noktalar içeren sekizli ağaç yapısındaki bir kapsayıcı nesnedir. Bu nesneler, sekizli ağaçlar gibi çarpışabilir, ölçülebilir ve tespit edilebilir [53].

## 3.3 HESAPLAMA MODÜLLERİ

CoppeliaSim yazılımı, bir veya daha fazla sahne nesnesi üzerinde doğrudan çalışabilen hesaplama modülleri ve fonksiyonları sunmaktadır. Yakınlık sensörü veya görüntü sensörü tarafından şekiller algılanırken hesaplama modüllerinin kullanılması gerekir. Temel hesaplama modülleri, kinematik modülü, dinamik modülü, çarpışma algılama modülü, mesafe hesaplama modülü ve yol planlama modülü olmak üzere beş gruba ayrılır [47].

### 3.3.1 Kinematik Modülü

Kinematik modülü, her türden mekanizma için ileri veya ters kinematik hesaplamalarını yapmaktadır. Sönümlü en küçük kareler yöntemiyle sözde ters (pseudoinverse) kinematik hesaplamayı kullanan bu modül, koşullu, sönümlü/sönümsüz ve ağırlıklı çözünürlüğü desteklemektedir [47].

### 3.3.2 Dinamik Modülü

Dinamik modülü, katı gövde dinamiği hesaplaması ile etkileşiminin yönetilmesini sağlar. Hesaplanan etkileşimler, nesneler veya modeller için çarpışma tepkisi ile kavrama gibi durumlarda kullanılır. Çeşitli hassasiyet, hız veya özellikler yardımıyla gerçekleştirilebilen fizik simülasyonlarının mümkün olduğunca gerçeğe yakın hâle getirilebilmesi için dinamik tabanlı fizik motorları devamlı bir gelişim içerisindedir. Çoğunlukla tahminlere dayanan fizik motorları için elde edilen sonuçların, farklı bir fizik motoru ile de kıyaslanması gerekebilir [47]. Bu nedenle CoppeliaSim yazılımı, Bullet Physics Library [59], Open Dynamics Engine [60], Vortex Studio [61] ve Newton Dynamics [62] fizik motorlarını destekleyerek simülasyon ihtiyaçlarına göre farklı bir motora istenildiği zamanda hızlıca geçişi kolaylaştırmaktadır [53].

### 3.3.3 Çarpışma Algılama Modülü

Çarpışma algılama modülü, çarpışabilir varlıklar arasında meydana gelebilecek çarpışmaları izlemeyi, kaydetmeyi ve görselleştirmeyi sağlar [53]. Şekiller veya şekil toplulukları arasında hızlı ve kesin bir girişim hesaplaması yapan bu modül, yalnızca çarpışmaları algılamaktadır. Çarpışma algılama modülü, dinamik modülün çarpışma tepkisi hesaplama algoritmasından tamamen bağımsız çalışır [47].

### 3.3.4 Mesafe Hesaplama Modülü

Mesafe hesaplama modülü, yalnızca mesafeleri ölçmekte kullanılır [53]. Bu modül, ölçülebilir varlıklar arasındaki en kısa mesafelerin izlenmesini, kaydedilmesini ve görselleştirilmesini sağlar. Şekiller veya şekil toplulukları arasındaki en kısa mesafeyi hızla ve kesin bir değerde hesaplar [47].

### 3.3.5 Yol Planlama Modülü

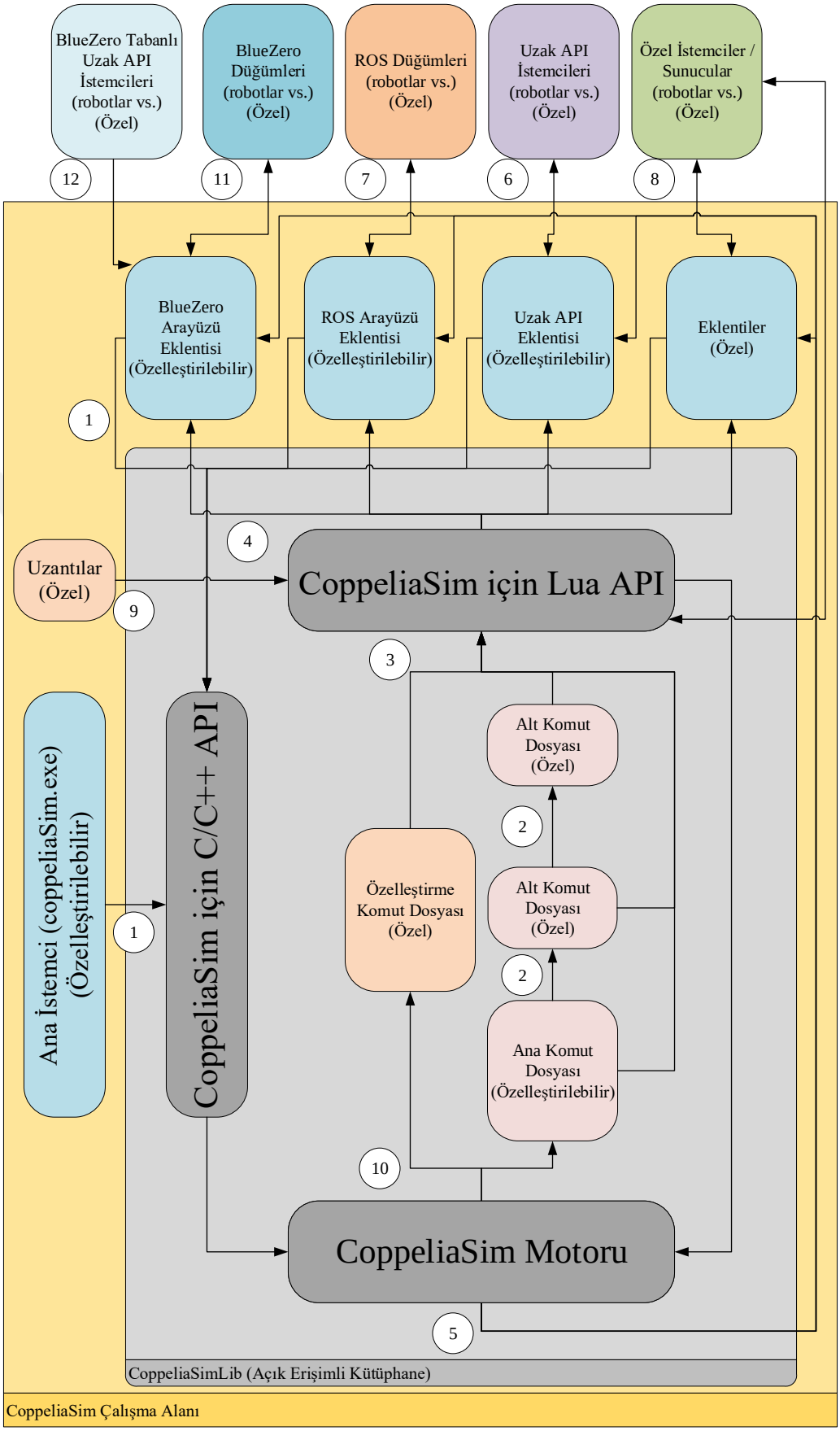
Yol planlama modülü, hareket ettiğinde bulunabileceği konumu mevcut konumuna bağlı (holonomik olmayan) ya da mevcut konumundan bağımsız (holonomik) sistemler için yol planlama görevlerini yürütür. Bunun için, Hızlı RRT (Rapidly-

exploring Random Tree, Keşfeden Rastgele Ağaçlar) algoritmasından [63] türetilen bir yaklaşım kullanır. Ayrıca kinematik zincirlerin yol planlama görevlerini de destekler [47].

## 3.4 SİMÜLASYON KONTROL MEKANİZMALARI

Simülasyon kontrol mekanizması, sahnede bulunan nesne ve modeller ile yazılım bileşenleri arasındaki iletişimi sağlayan bir kontrol sistemi bütünüdür. Bir simülasyon senaryosunu oluşturmak için kodlar hazırlanırken kullanılan komutların nesnelere göre alt bölümlere ayrılarak dağıtılması daha etkili bir kontrol sağlar. Çoğaltma işleminin basitleştirilmesi ve çoklu robot simülasyonlarının kolaylaştırılabilmesi için bölümlere ayırma gereklidir. Ayrılan bölümler, simülasyonu hızlandırırken aynı zamanda modellerin yerel bir ana kod üzerinden kontrol edilebilmesine de imkân tanımaktadır [47].

CoppeliaSim yazılımı, aynı anda birlikte çalışabilen ve birbiriyle uyumlu farklı programlama tekniklerine sahiptir [47]. Şekil 3.2'de CoppeliaSim yazılımının kontrol mimarisinin diyagramı gösterilmektedir. Bu şekilde yer alan (1) numaralı kısım C/C++ API çağrılarını, (2) numaralı kısım basamaklı alt komut dosyası yürütmeyi, (3) numaralı kısım Lua API çağrılarını, (4) numaralı kısım özel Lua API geri çağrılarını, (5) numaralı kısım CoppeliaSim olay geri çağrılarını, (6) numaralı kısım uzak API fonksiyon çağrılarını, (7) numaralı kısım ROS geçişini, (8) numaralı kısım özel iletişimi (soket, seri haberleşme bağlantısı, kanal hatları vb.), (9) numaralı kısım uzantı çağrılarını, (10) numaralı kısım CoppeliaSim özelleştirilmiş komut dosyası yürütme çağrılarını, (11) numaralı kısım BlueZero veri transferini ve (12) numaralı kısım BlueZero temelli uzak istemci çağrılarını göstermektedir.




Şekil 3.2. CoppeliaSim yazılımının kontrol mimarisini gösteren şematik diyagram.

### 3.4.1 Gömülü Komut Dosyaları (Embedded Scripts)

CoppeliaSim yazılımının diğerlerinden ayıran en güçlü özelliği, gömülü komut dosyalarını çalıştırabilmesidir. Gömülü komut dosyaları, bir eklemin temel düzeyde özelleştirilerek denetlenmesinde kullanılabilen geri çağırma komutlarını içermektedir. Gömülü komut dosyalarının işlevselliği, Lua programlama dilinde yazılan ek kitaplıklar kullanılarak ya da eklenti aracılığıyla kaydedilen özel fonksiyonlar kullanılarak genişletilebilmektedir. CoppeliaSim yazılımı, desteklediği çeşitli programlama tekniklerini birbirine bağlamak için gömülü komut dosyalarını kullanmaktadır [47].

Gömülü komut dosyaları, ana komut dosyası ve alt komut dosyaları olmak üzere iki gruba ayrılır. Ana komut dosyası (main script), Lua programlama dilinde basit bir betiktir. Bu betik, bir simülasyon sahnesindeki genel işlevlerle ilgilenen temel parçadır. Kinematik veya dinamik işlemler için farklı fonksiyonları çağırabilen ana komut dosyası, sahne hiyerarşisine göre basamaklı bir şekilde alt komut (child script) dosyalarını çağırmaktan da ayrıca sorumludur [47].

Alt komut dosyası, sadece belirli bir nesneye eklenerek simülasyonun ilgili bölümünü işletir. Bu komut dosyası, ait olduğu nesnenin ayrılmaz bir parçasıdır ve nesneyle birlikte çoğaltılabilir. CoppeliaSim yazılımında oluşturulan model dosyası, model tanımının yanı sıra modelin kontrolünü veya işlevlerini de içerebilmektedir. Model dosyaları, değiştirilerek çoğaltılabilmekte ve derlemeye gerek olmadan simülasyonda yeniden kullanılabilmektedir. Alt komut dosyaları ROS yayıncılarını/abonelerini kaydedebilmekte, iletişim hatlarını (soket, seri haberleşme bağlantısı vb.) açıp işleyebilmekte, yürütülebilir dosyaları açabilmekte, eklentileri yükleyebilmekte/kaldırabilmekte veya uzak API sunucu hizmetlerini başlatabilmektedir [47].

Alt komut dosyaları, iş parçacıklı ya da iş parçacıklı olmayan bir şekilde yürütülebilir. CoppeliaSim yazılımının iş parçacığı zamanlayıcısı, işlediği iş parçacıklarının eş yordamlar olarak davranmalarını ve görünmelerini sağlar. Bu zamanlayıcı, iş parçacıklarının yürütme sırasını öne ya da sona alarak kontrol eder. Bu sayede, ana

komut dosyası ile diğer alt komut dosyalarını eşzamanlı hâle getirir. Ayrıca her iş parçacığı serbest yürütülecek şekilde programlanarak geçici süreyle asıl iş parçacığı gibi çalışması sağlanabilir [47].

### 3.4.2 Uzantılar (Add-ons)

Uzantılar, kullanıcı tarafından Lua programlama dilinde yazılan fonksiyonlarla yazılım işlevselliğinin genişletilmesini sağlayan ve yazılımın başlangıcında otomatik olarak yüklenen araçlardır. CoppeliaSim yazılımı, bağımsız fonksiyonlar ve komut dosyaları olmak üzere iki tür uzantıyı destekler. Bağımsız uzantı fonksiyonları, içe veya dışa aktarıcı olarak uygulanabilen ve kullanıcı tarafından seçildiğinde bir kez yürütülen fonksiyonlardır. Uzantı komut dosyaları ise, arka planda sürekli olarak etkin bir şekilde çağırılan komut dosyalarıdır. Bu komut dosyaları, simülasyonun yavaşlatılmaması için sadeci (minimalistic) kodlara sahiptir. Simülasyon yazılımının özelleştirilmesinde kullanılan uzantı komut dosyaları, korumalı alanın (sandbox) komut dosyasıyla birçok özelliği paylaşır [53].

### 3.4.3 Eklentiler (Plug-ins)

Eklenti, kullanıcı tarafından yazılan fonksiyonlar ile yazılım işlevselliğinin genişletilmesine imkân veren paylaşımlı bir kitaplıktır. Belirli bir simülasyon modeli veya sahne nesnesini özelleştirmek için eklenti kullanılır. CoppeliaSim yazılımında bir eklenti, ana istemci uygulaması tarafından otomatik olarak yüklenebileceği gibi sim.loadModule/sim.unloadModule aracılığıyla dinamik olarak da yüklenebilir veya kaldırılabilir. Eklentinin programlama dili, dışa aktarılmış C fonksiyonlarını çağırabilen ve paylaşımlı bir kitaplık oluşturabilen herhangi bir dil olabilir. Yazılıma hızlı hesaplama yeteneği kazandırmada veya başka bir donanıma arayüz oluşturmada genellikle eklentiler tercih edilir. Eklentiler, özellikle başka dilde yazılmış kodları çalıştırması için bir sarmalayıcı (wrapper) görevi üstlenebilmektedir. Örneğin Atmel mikrodenetleyicilerinde yazılan kodu işleyen ve yürüten bir eklenti bulunmaktadır [53]. Ayrıca yazılımın uzak API arabirimi ve ROS arabirimi de eklentiler aracılığıyla uygulanmaktadır [47].

### 3.4.4 Uzak API İstemcileri (Remote API Clients)

Uzak API arabirimi, bir sahnenin uzaktan yüklenmesi veya bir simülasyonun uzaktan başlatılması, duraklatılması ya da durdurulmasında kullanılabilir. Bu arabirim, soket iletişimi ya da paylaşılan bellek yoluyla haricî bir uygulama ile yazılımın etkileşime geçmesini sağlar. Çapraz platform desteğine sahip olan uzak API arabirimi, çift yönlü hızlı veri akışına izin verir. BlueZero düğümlerine kıyasla daha az bağımlılık gösteren uzak API istemcileri, genişletilmesi zor olduğu için kullanışlı değildir [53].

Uzak API arabirimi, uzak API sunucu hizmetleri ve uzak API istemcileri tarafından oluşturulur. İstemci tarafı, gerçek robotlar da dâhil olmak üzere çoğu donanıma C/C++, Python, Java, MATLAB, Octave ve Lua programlama dillerinde küçük bir ayak izi kodu olarak yerleştirilebilir. Sunucu tarafı ise, yazılım tarafından varsayılan olarak yüklenen bir eklenti (simExtRemoteApi.dll, libsimExtRemoteApi.dylib, libsimExtRemoteApi.so) aracılığıyla uygulanır. İşlem modu, engelleyici (sunucu cevap verene kadar bekler) veya engellemeyici (akan komutları bir arabellekten okur veya sunucu tarafında bir akış hizmetini başlatır/durdurur) fonksiyonların çağrılmasını sağlar [47].

### 3.4.5 ROS Düğümleri (ROS Nodes)

CoppeliaSim yazılımı, komutlarının ROS hizmetleri aracılığıyla çağırılmasına veya ROS yayıncıları/aboneleri aracılığıyla veri akışında bulunabilmesine imkân tanıyan ROS arayüzü eklentisine sahiptir. Bu eklenti, ROS yayıncıları/aboneleri, servis çağrısı veya doğrudan yazılım içerisindeki bir gömülü komut dosyası talimatıyla etkinleştirilebilir [47].

### 3.4.6 BlueZero Düğümleri (BlueZero Nodes)

BlueZero (BØ) yöntemi, haricî bir uygulamanın CoppeliaSim yazılımı ile bağlantı kurmasına izin verir. Uzak API istemcisi olarak da kullanılabilen BØ, birden çok iş parçacığında, birden çok işlemde ve hatta birden çok makinede çalışan yazılım parçalarını birbirine bağlayan araçları oluşturur. Bir çapraz platform ara yazılımı olan

BØ için güncel sürüm, C++, Java, Python, MATLAB ve Lua programlama dillerini desteklemektedir. BØ arayüzünün ROS arayüzü ile benzer yanları bulunmaktadır. BØ, yalnızca iletişim paradigmaları (istemci/sunucu, yayıncı/abone) ve mesaj aktarımı (ZeroMQ dayalı) sağlamaya odaklanmakla birlikte kullandığı ortak protokoller, veri yapıları ve mesaj serileştirme formatı açısından agnostik bir işleyişe sahiptir [53].

ØMQ (Zero Message Queue, Sıfır Mesaj Kuyruğu) olarak kısaltılabilen ZeroMQ, gömülebilir bir ağ kitaplığı gibidir. İşlem içi, işlemler arası, TCP (Transmission Control Protocol, İletim Kontrol Protokolü) ve çok noktaya yayın (multicast) gibi çeşitli aktarımlar için atomik mesajlar taşıyan soketler kullanan ØMQ, eşzamanlılık çerçevesi gibi davranmaktadır. İsminde bulunan sıfır, aracı (broker), gecikme, yönetim, maliyet ve atık gibi hedeflerde sadeciliğin ön plana çıkarıldığını ifade etmektedir [64].

## 3.5 KOMUT DOSYALARI

CoppeliaSim yazılımı, her simülasyon nesnesinin davranışını yönetmek için çeşitli kontrol mekanizmaları sunar. Bu kontrol mekanizmaları simülasyon ortamının içinden veya dışından uygulanabilmektedir. Temel iç kontrol mekanizması, sahnedeki herhangi bir öğeyle ilişkilendirilebilen alt komut dosyalarının kullanılmasıdır. Bu alt komut dosyaları, ilişkili oldukları nesnelerin ayrılmaz bir parçası olup simülasyonun belirli bir bölümünü işletir. CoppeliaSim yazılımının kendine özgü model dosyası sayesinde, nesne modelleri ile birlikte kontrol parametreleri ve fonksiyonları taşınabilmekte, ölçeklenebilmekte ve çoğaltılabilmektedir [49].

### 3.5.1 Ana Komut Dosyası (Main Script)

Ana komut dosyası, sistem tarafından uygun şekilde çağrılan geri çağırma fonksiyonlarını içerir. Varsayılan ana komut dosyası, başlatma, hareket ettirme, algılama ve restorasyon olmak üzere dört temel sistem geri çağırma fonksiyonuna bölünmektedir. Başlatma fonksiyonu dışında kalan diğer tüm fonksiyonlar isteğe bağlı olarak çalıştırılmaktadır. Tanımlanmamış sistem geri çağırma fonksiyonları için yapılan tüm çağrılar göz ardı edilmektedir [53].

#### 3.5.1.1 Başlatma Fonksiyonu (Initialization Fuction)

Başlatma fonksiyonu (sysCall_init), simülasyonu hazırlamaktan sorumludur. Bu sistem geri çağırma bölümü, isteğe bağlı olarak düzenlenemediği gibi simülasyon çalıştırıldığında yalnızca bir kez yürütülür [53].

#### 3.5.1.2 Hareket Ettirme Fonksiyonu (Actuation Function)

Hareket ettirme fonksiyonu (sysCall_actuation), simülasyonu devamlı olarak çalıştırır. Her simülasyon geçişinde yürütülen bu bölüm, ters kinematik ve dinamik gibi simülasyondaki tüm hareket ettirme işlevselliğini genel olarak ele almaktan sorumludur. Hareket ettirme bölümü, iş parçacıklı alt komut dosyalarını başlatan (sim.launchThreadedChildScripts), iş parçacıklı alt komut dosyalarını devam ettiren (sim.resumeThreads) ve iş parçacığı olmayan alt komut dosyalarına ait hareket ettirme fonksiyonlarını çağıran (sim.handleChildScripts) üç önemli komuta sahiptir. Bu komutlar olmadan, alt komut dosyaları ya da alt komut dosyalarının hareket ettirme fonksiyonları yürütülemez ve özelleştirilmiş model işlevleri istendiği gibi çalıştırılamaz [53].

Hareket ettirme bölümünde, sadece iş parçacıklı alt komut dosyalarının durumu takip edilmektedir. Bu sayede sahne nesneleri ve alt komut dosyaları, simülasyon çalışırken herhangi bir anda kopyalanıp sahneye yapıştırılarak çoğaltılabilmektedir. Simülasyonun sonlandırılabilmesi için iş parçacıklı alt komut dosyalarının yürütülmesi tamamlanmalıdır. Aksi takdirde, yazılım simülasyonu sonlandırmadan önce gerekenden daha uzun süre bekleyerek restorasyon bölümünü hemen çalıştırmayabilir [65].

#### 3.5.1.3 Algılama Fonksiyonu (Sensing Function)

Algılama fonksiyonu (sysCall_sensing), simülasyon yazılımının yakınlık sensörleri ve çarpışma algılama modülü gibi tüm algılama işlevlerini genel olarak ele almaktan sorumludur. Her simülasyon geçişinde yürütülen bu bölümde, iş parçacıklı alt komut dosyalarını sürdüren (sim.resumeThreads) ve iş parçacıklı olmayan alt komut

dosyalarının algılama fonksiyonlarını çağıran (sim.handleChildScripts) iki komut önemlidir. Bu komutlar olmadan, alt komut dosyaları ya da alt komut dosyalarının algılama fonksiyonları yürütülemez ve belirlenen model davranışları istendiği gibi çalıştırılamaz [53].

#### 3.5.1.4 Restorasyon Fonksiyonu (Restoration Function)

Restorasyon fonksiyonu (sysCall_cleanup), simülasyon bitmeden hemen önce bir kez çalıştırılan ana komut dosyasının son kısmıdır [65]. Bu bölüm, nesnelerin ilk yapılandırmasını geri yükleme, sensör durumlarını ve çarpışma durumlarını temizleme gibi işlemlerden sorumludur [53].

### 3.5.2 Alt Komut Dosyaları (Child Scripts)

Ana komut dosyası, yürütülmesi iş parçacıklı ve iş parçacıklı olmayan iki farklı türdeki alt komut dosyalarını çalıştırmaktadır. Alt komut dosyaları, yürütülebilir dosyaları başlatma, uzak API sunucularını başlatma, iletişim hatlarını açma/yönetme, eklentileri yükleme/kaldırma ve ROS/BlueZero yayıncıları ile abonelerini kaydetme gibi birçok farklı görevde kullanılabilir [49].

#### 3.5.2.1 İş Parçacıklı Olmayan Alt Komut Dosyaları (Non-Threaded Child Scripts)

İş parçacıklı olmayan alt komut dosyaları, her çağrıldıklarında bazı görevleri yerine getirir ve ardından kontrol olarak çağırıldığı betiğe geri döner. Fonksiyon olarak çalışan bu alt komut dosyaları, her simülasyon adımında farklı bir simülasyon süresi ile çağrılırlar. Doğrudan geçiş yapılan alt komut dosyası, kontrolü çağıran betiğe geri dönmediği takdirde tüm simülasyon durdurulmaktadır [65].

#### 3.5.2.2 İş Parçacıklı Alt Komut Dosyaları (Threaded Child Scripts)

İş parçacıklı alt komut dosyaları, bir iş parçacığı içerisinde başlatılır ve ana komut dosyasının kodu tarafından yürütülür. Bu türde bir alt komut dosyasına yapılan çağrı,

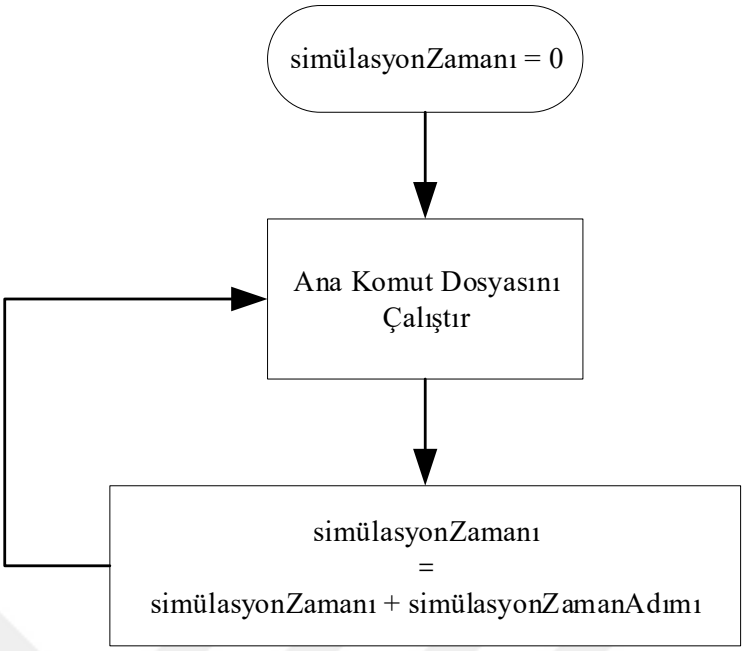
yeni bir iş parçacığı başlatıp betiğe geri döner. Başlatılan yeni iş parçacığı ise çalışmasına devam eder ve paralel olarak işletilir. Önceden başlatılmış bir iş parçacıklı alt komut dosyasının çalışması henüz sona ermeden tekrar çağrılırsa yeni çağrı yok sayılır [65].

Tüm iş parçacıklı alt komut dosyaları, varsayılan olarak bir kez çalıştırılacak şekilde ayarlanmıştır. İhtiyaca göre, bir iş parçacığı alt komut dosyası çok sayıda da çalıştırılabilmektedir. Ayrıca iş parçacıklı alt komut dosyaları, diğer iş parçacıklı alt komut dosyalarını çağırabildikleri gibi uygun olduğu takdirde iş parçacıklı olmayan alt komut dosyalarını da çağırabilir. İş parçacıklı olmayan bir alt komut dosyası, ana iş parçacığından farklı bir iş parçacığı tarafından çağrılması durumunda zaten bir iş parçacığı içerisinde çalıştığını bildirecektir [65].

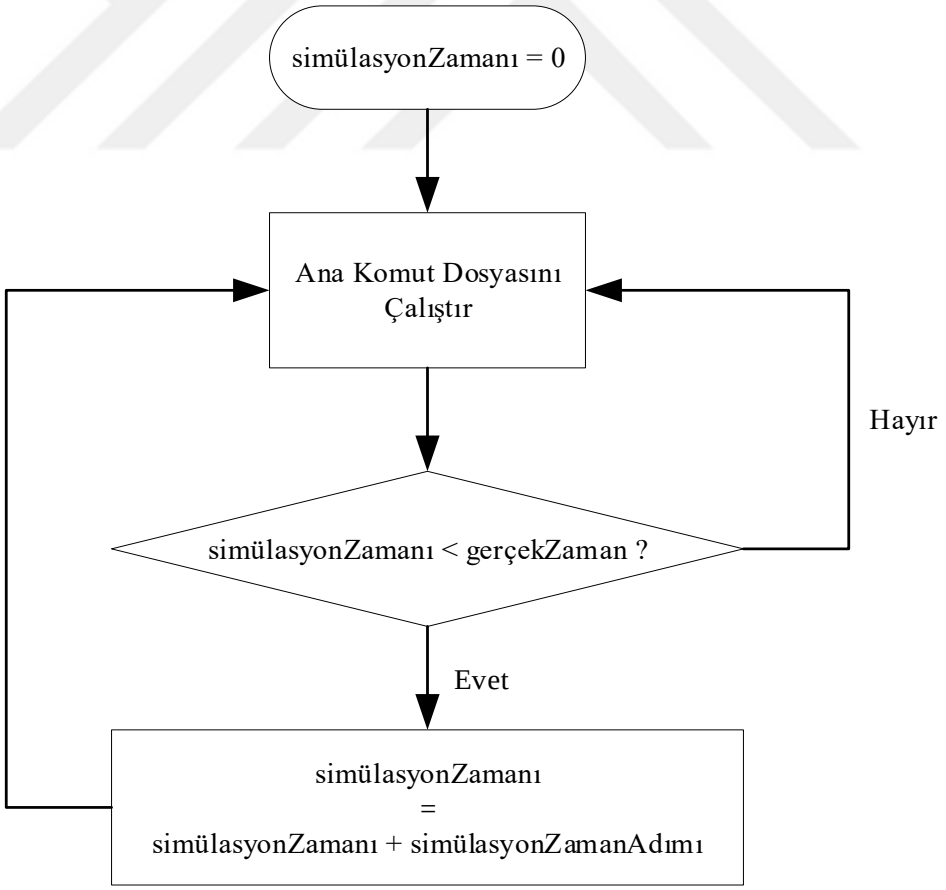
İş parçacıklı alt komut dosyaları, iş parçacıklı olmayan alt komut dosyalarına kıyasla daha ileri programlama bilgisi gerektirir. Uygun şekilde programlanmayan bir iş parçacıklı alt komut dosyası, daha fazla işlem gücüne ihtiyaç duyabilir, işlem süresini boşa harcayabilir ve simülasyon durdurma komutuna cevap vermekte gecikebilir [49].

## 3.6 SİMÜLASYON SÜRECİ

CoppeliaSim yazılımında simülasyon zamanı, ana komut dosyası her yürütüldüğünde bir simülasyon zaman adımı ile artırılır. Varsayılan ana çalışma modunda Şekil 3.3 üzerinde gösterildiği gibi simülasyon süresi sabit zaman adımlarıyla ilerletilmektedir. Bu zaman adımının uzatılması, hızlı ancak yanlış ve istikrarsız simülasyona neden olmaktadır. Ayrıca anlık ağır hesaplamalar sonucunda simülasyon zamanının gerçek zamanı takip edememesi durumuyla da karşılaşılabilir. Buna çözüm olarak CoppeliaSim yazılımına gerçek zaman modu eklenmiştir. Gerçek zaman modu etkinleştirildiğinde, Şekil 3.4 üzerinde gösterildiği gibi simülasyon zamanı kayıp zamanı yakalamaya çalışır. Bu simülasyon modu, simülasyonun karmaşıklığına, bilgisayarın performansına ve simülasyon ayarlarına bağlı olarak her zaman istenildiği gibi davranmayabilir [65].




Şekil 3.3. Ana simülasyon döngüsü.




Şekil 3.4. Gerçek zamanlı simülasyon döngüsü.

## 3.7 NESNE MODELLERİNİN EKLENMESİ

CoppeliaSim yazılımı, mekanik modelin oluşturulmasına veya değiştirilmesine imkân tanır. Bilgisayar destekli tasarım yazılımında bir araya getirilen tüm model parçaları, bir CAD dosyası üzerinden CoppeliaSim yazılımına aktarılabilir ve ayrı mekanik özellikleri tanımlarken yeniden alt parçalara bölünebilir. Yazılıma aktarılan CAD dosyaları üzerinden simülasyon için basitleştirilmiş dinamik modeller oluşturulabilir. Aktarılan bir nesne için uygun bir kütle ve atalet momenti değeri ayarlanarak katı gövde dinamik özellikleri belirlenebilir [49].

CoppeliaSim yazılımı, üç boyutlu nesneleri tanımlayan .dxf, .ply, .dae, .obj ve .stl dosya türlerini içe aktarma özelliğine sahiptir. Bu dosya türlerinden birini kullanan CAD modeli, genellikle tek parça katı blok nesnesi olarak içe aktarılır. Yazılımda, içe aktarılan CAD bloğunun uygun olarak parçalara ayrılabilmesini sağlayan araçlar bulunmaktadır. Öte yandan bilgisayar destekli tasarım yazılımının verileri oluşturma şekline bağlı olarak CAD örgü dosyası birkaç parçaya bölünmüş veya farklı ölçek ve konumda aktarılabilmektedir [51]. Bir nesne için, model ağının simülasyonu yavaşlatacak kadar çok sayıda üçgen içerip içermediğinin kontrol edilmesi gerekir. Modelin çok sayıda üçgen içermesi durumunda, üçgen sayısını azaltan örnek seyreltme (desimasyon) aracı kullanılabilir. Ancak bu araç kullanıldığında, asıl parçanın deforme olacağı unutulmamalıdır. Son olarak, parçaları birbirine bağlayarak birbirleriyle olan ilişkilerini belirleyen eklemlerin aktarılan nesneye uygulanması ve her parça için uygun atalet momenti değerlerinin girilmesi gerekir [66].

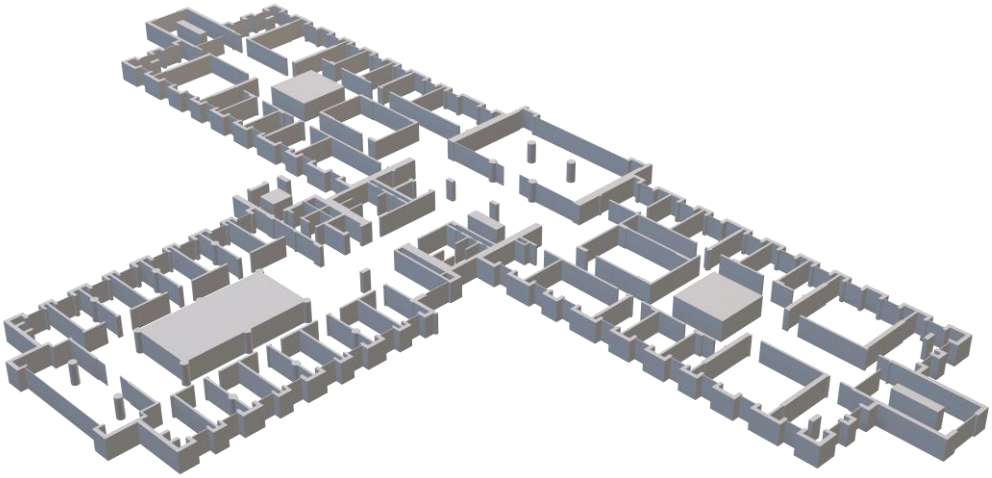
### 3.7.1 Tekerlekli Sandalye

Bu tez çalışmasında hazırlanan akıllı tekerlekli sandalye sisteminin genel geometrisi, Şekil 3.5'te gösterildiği gibi sıradan bir tekerlekli sandalye modeli ile basitleştirilmiştir. Bu basitleştirilmiş nesne, genel sistem dinamikleri aynı tutulmaya çalışılırken simülasyonun hesaplama yükünün azaltılabilmesi için CoppeliaSim yazılımında kullanılmıştır. Ardından simülasyondaki tekerlekli sandalye nesnesi için her parçaya atalet momenti değerleri girilmiş, tekerlere eklemler uygulanmış ve hareket edebilmesini sağlayan gömülü komut dosyası eklenmiştir.

Şekil 3.5. Simülasyonda kullanılan sıradan bir tekerlekli sandalye modeli.

### 3.7.2 Kat Planı

Akıllı tekerlekli sandalye sisteminin test edilmesi için Karabük Üniversitesi Mühendislik Fakültesi binasının dördüncü katı kullanılmıştır. Bu ortamın Şekil 3.6'da gösterilen üç boyutlu modeli, iki boyutlu kat planı çizimine yükseklik boyutunun eklenmesiyle oluşturulmuştur. Ardından kat planının üç boyutlu modeli CoppeliaSim yazılımının simülasyon sahnesine aktarılmıştır. Akıllı tekerlekli sandalye için hazırlanan görsel arayüz yazılımı, gerçek ortamda denenmeden önce CoppeliaSim yazılımında test edilmiştir.



Şekil 3.6. KBÜ Mühendislik Fakültesi binasının dördüncü katının modellenmesi.

Haritalama işlemi akıllı tekerlekli sandalyenin doğru olarak konumlandırılması için önem arz etmektedir. Uygun bir oranda ölçeklenerek elde edilen simülasyon ortamına tespit edilebilir nesnelerden yürüyen insan modeli eklenmiştir. Akıllı tekerlekli sandalye sistemi için hazırlanan arayüz yazılımının sadece durağan elemanlara göre davranmayıp hareketli elemanlar için engel tespiti de yaparak yol bilgilerini sürekli güncellemesi gerekir. Bu sebeple, temelde yer alan statik haritaların iki boyutlu çizimleri, derinlik algılayıcı kamera üzerinden elde edilen verilerin yorumlanması ile birleştirilerek anlık değişebilen dinamik haritalara dönüştürülmektedir.

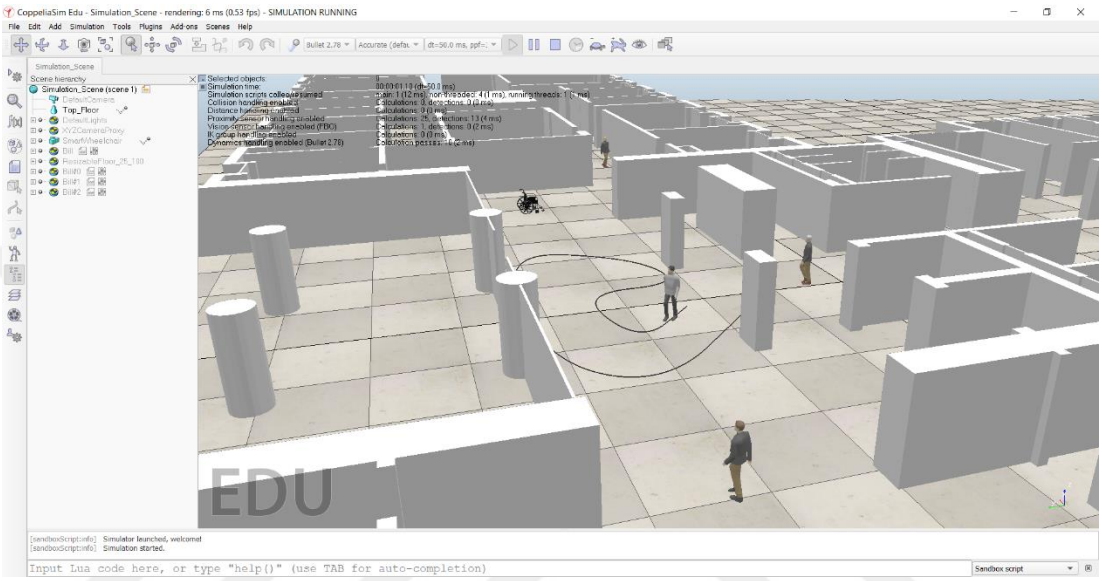
## 3.8 SİMÜLASYON MODELİNİN ARAYÜZ YAZILIMI İLE ETKİLEŞİME GEÇEREK KULLANILMASI

CoppeliaSim yazılımında simülasyonun kontrol edilmesi için farklı dillerde programlama yöntemleri kullanılabilmekte, sistem doğrulama, algoritma optimizasyonu, endüstriyel otomasyon uygulamaları, karmaşık montaj zincirlerinin simülasyonu, robot görev planlayıcıları ve denetleyicileri gibi çeşitli görevleri gerçekleştirebilmektedir [47]. Karmaşık ortam simülasyonu için sunduğu dağıtılmış kontrol yaklaşımı sayesinde, CoppeliaSim yazılımı simülasyonları çok yönlü ve ölçeklenebilir hâle getirmiştir [65].

CoppeliaSim yazılımındaki bir simülasyon modeli, alt komut dosyası, ROS/BlueZero düğümleri, uzak API temelli haricî istemci uygulamaları veya yazılan eklentiler üzerinden çeşitli yollarla kontrol edilebilir. Yazılımın eklentisi veya komut dosyası ile kanal hatları (pipelines), soketler (sockets) ya da seri haberleşme bağlantısı (serial port) aracılığıyla iletişim kuran haricî bir uygulama yazılabilmektedir. Bu tez çalışmasında, görsel arayüz yazılımı ile simülasyonun bağlantı kurması sanal seri haberleşme bağlantıları üzerinden sağlanmıştır.

Güzergâh belirleme fonksiyonlarının test edilmesi için CoppeliaSim yazılımında zaman aralığı ve dinamik motorun uygun olarak ayarlanması sonucunda gerçek zamanlı simülasyon gerçekleştirilmektedir. Yazılım içerisinde aynı ya da farklı modellerin çok farklı yapılandırmalarla yeniden test edilmesi mümkündür. Böylece, oluşturulan akıllı tekerlekli sandalye sisteminin güzergâh planlamasında verdiği

tepkiler farklı ortamlarda hızlıca test edilebilmektedir. Ayrıca nesneler, daha kaotik ve rastgele hareket eğilimleri kazandırılarak dinamik bir şekilde etkileşime geçirilebilmektedir [48]. Bu özellik, Şekil 3.7'de gösterildiği gibi hem planlı hem de rastgele hareket eden insanlar simülasyona eklenerek çarpışma algılama modülünün denenmesi için kullanılmıştır.




Şekil 3.7. CoppeliaSim yazılımında simülasyonun çalıştırılması.

# BÖLÜM 4

# YOL BULMA ALGORİTMASININ BELİRLENMESİ

Güzergâh belirleme problemlerinde genellikle ızgara tabanlı bir grafik üzerinde sunulan çevre ortamı temsilî için başlangıç konumdan bitiş konumuna bir yol planlama görevi verilir. Grafikteki ızgara hücreleri (karolar), ortama bağlı olarak bir engel veya boş alan içeren ikili (binary) değerde olabileceği gibi ilgili alanın geçme zorluğunu yansıtan bir değerle de ilişkilendirilebilir. İşlemde kullanılan algoritmaya göre belirlenen ızgara hücrelerinin merkezine ilgili konumu temsil eden bir düğüm yerleştirilir. Izgara hücreleri arasında ilişki kurulması için kullanılan komşuluk bağlantısı ise bitişik hücreler arasındaki kenar veya köşeler üzerinden oluşturulur [67].

Başlangıç ve bitiş noktası girildikten sonra sistem bir güzergâh aramaya başlayabilir. Güzergâh arama süreci boyunca bulunan komşuların sayısı hızla artacağından, düğümlerin olabildiğince erken dışlanabilmesi için bir arama stratejisi gerekir. Bu, her bağlantı için bir maliyet tanımlanarak sağlanabilir [68].

Aramaya dayalı yol bulma algoritmaları, ileri arama, geri arama ve çift yönlü arama yöntemlerinden biri ile yol araştırır. Bu algoritmalar, işlem süreci boyunca bir ağaç oluşturur. Ağacın kök düğümü, ileri arama için başlangıç düğümüyken geri arama için bitiş düğümüdür. İleri arama algoritmaları, aynı yol bulma problemi için bitiş durumundan başlangıç durumuna geçilmesiyle geri arama yöntemine dönüştürülebilir. Gereken keşfin azaltılması için ileri ve geri aramanın birleştirilmesi ile de çift yönlü arama oluşturulabilir [69].

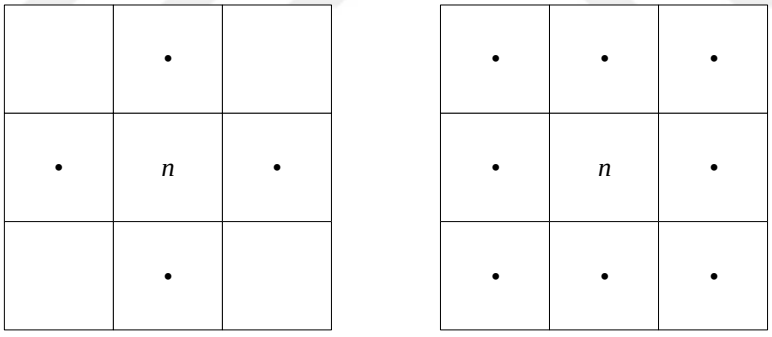
İleri arama yöntemini kullanan algoritmalara, sığ öncelikli arama (Breadth-First Search, BFS), derinlik öncelikli arama (Depth-First Search, DFS), en iyi öncelikli arama (Best-First Search), yinelemeli derinleşme (Iterative Deepining), Dijkstra,

A* (A-Yıldız, A-Star), D* (D-Yıldız, D-Star) ve atlama noktası arama (JPS) örnek olarak verilebilir [69].

Bu bölümde, yol planlama alanındaki A* algoritması ve bu algoritmanın iyileştirmesi olan atlama noktası araması tanıtılmaktadır. Algoritma tanıtımından önce düğüm ve komşular arasındaki uzaklık ölçütlerine değinilecektir.

## 4.1 DÜĞÜM İLE KOMŞULARI ARASINDAKİ UZAKLIK ÖLÇÜTLERİ

Kare hücreli ızgara tabanlı bir grafikte düğümler arası komşuluklar, görüntü işlemede kullanılan pikseller arasındaki komşuluklara benzetilerek ele alınabilir. Komşuluk, Şekil 4.1'de göründüğü gibi dörtlü, sekizli veya daha fazlası olabilir [70]. Dörtlü komşulukta, her düğümün yukarıda, aşağıda, sağda ve solda olmak üzere (düz yönlerde) dört adet bağlı komşusu vardır. Sekizli komşulukta ise, dörtlü komşuluğa ek olarak çapraz yönlerde temas eden diğer dört düğüm de komşu olarak kabul edilmektedir.




(a) (b)

Şekil 4.1. Bir düğümün (node) komşuları. a) Dörtlü komşulukta bir düğümün (*n*) komşuları (•). b) Sekizli komşulukta bir düğümün (*n*) komşuları (•).

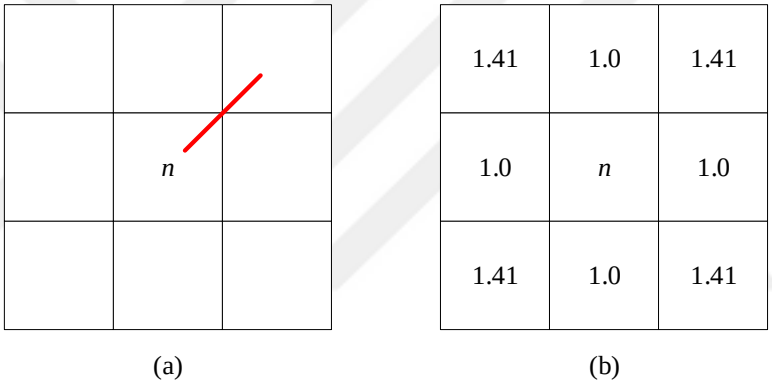
Komşu düğümler arasındaki mesafenin hesaplanabilmesi için birkaç farklı uzaklık ölçütü bulunmaktadır. Bunlardan en önemlileri Öklid uzaklığı, şehir blok uzaklığı ve satranç tahtası uzaklığıdır.

### 4.1.1 Öklid Uzaklığı (Euclidean Distance)

Öklid uzaklığı, iki düğüm arasındaki düz çizgi mesafesi olarak tanımlanmaktadır [71]. Sırasıyla koordinatları $(x_1, y_1)$ ve $(x_2, y_2)$ olan $a$ ve $b$ düğümleri için aralarındaki Öklid mesafesi Eşitlik 4.1 ile hesaplanmaktadır.

$$d_e(a,b) = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2} \quad (4.1)$$

Bir düğüm ile sekizli komşuluktaki komşu düğümleri arasındaki Öklid mesafeleri Şekil 4.2'de gösterilmiştir.




Şekil 4.2. Kare hücreli ızgaradaki bir düğüm için Öklid mesafeleri. a) Geçerli *n* düğümü ve sekizli komşuluktaki komşuları. b) Komşu düğümlerin geçerli *n* düğümüne olan Öklid mesafe değerleri.

### 4.1.2 Şehir Bloğu Uzaklığı (City Block Distance)

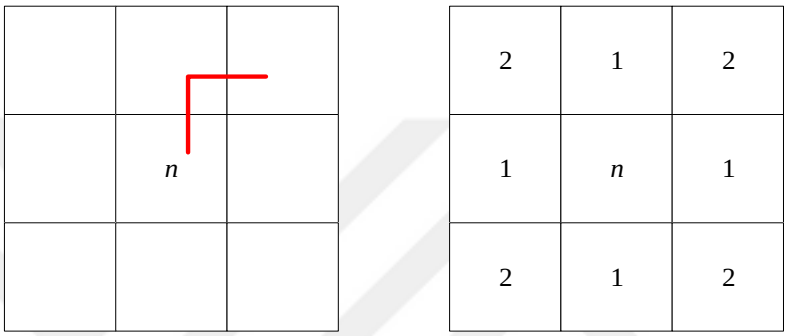
Manhattan veya taksi (taxicab) mesafesi olarak da bilinen şehir bloğu uzaklığında, iki düğüm arasındaki mesafe dörtlü komşuluğa göre ölçülmektedir. Çapraz komşular, yatay ve dikey adım gerektirdiğinden iki mesafelidir. Kenarları birbirine değen komşular 1 birim aralıktayken, köşeleri birbirine değen komşular 2 birim aralıktadır.

Şehir bloğu uzaklığında bir düğümden diğerine geçerken, yalnızca düğüm ızgarasına paralel çizgiler boyunca ortogonal hareketin mümkün olduğu kabul edilir. Çapraz hareketlere izin verilmez. Buna göre, sırasıyla koordinatları $(x_1, y_1)$ ve $(x_2, y_2)$ olan

$a$ ve $b$ düğümleri arasındaki şehir bloğu uzaklığı Eşitlik 4.2 ile hesaplanmaktadır [71].

$$d_m(a,b) = |x_1 - x_2| + |y_1 - y_2| \quad (4.2)$$

Bir düğüm ile sekizli komşuluktaki komşu düğümleri arasındaki şehir bloğu mesafeleri Şekil 4.3'te gösterilmiştir.




(a) (b)

Şekil 4.3. Kare hücreli ızgaradaki bir düğüm için şehir bloğu mesafeleri. a) Geçerli *n* düğümü ve sekizli komşuluktaki komşuları. b) Komşu düğümlerin geçerli *n* düğümüne olan şehir bloğu mesafe değerleri.

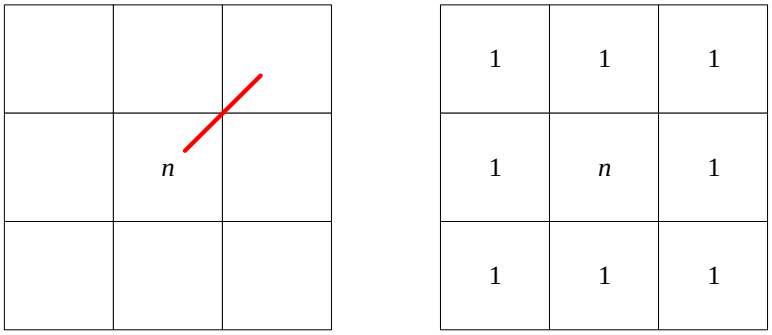
### 4.1.3 Satranç Tahtası Uzaklığı (Chessboard Distance)

Satranç tahtası uzaklığı, satrançta bir kralın kullanabileceği hareketlere benzetilir ve Chebyshev mesafesi olarak da bilinir [70]. Bu uzaklık ölçütünde, sekizli komşuluğa göre kenarları veya köşeleri birbirine temas eden düğümler 1 birim aralıktadır.

Satranç tahtası uzaklığında hem düz hareketlerin hem de çapraz hareketlerin yapılabileceği kabul edilir. Buna göre, sırasıyla koordinatları $(x_1, y_1)$ ve $(x_2, y_2)$ olan $a$ ve $b$ düğümleri arasındaki satranç tahtası uzaklığı Eşitlik 4.3 ile hesaplanmaktadır [71].

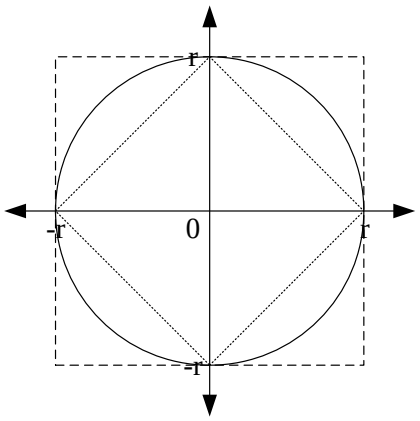
$$d_c(a,b) = \max\left(|x_1 - x_2|, |y_1 - y_2|\right) \quad (4.3)$$

Bir düğüm ile sekizli komşuluktaki komşu düğümleri arasındaki satranç tahtası mesafeleri Şekil 4.4'te gösterilmiştir.




Şekil 4.4. Kare hücreli ızgaradaki bir düğüm için satranç tahtası mesafeleri. a) Geçerli *n* düğümü ve sekizli komşuluktaki komşuları. b) Komşu düğümlerin geçerli *n* düğümüne olan satranç tahtası mesafe değerleri.

Şekil 4.5'te gösterilen düz çizgi Öklid uzaklığına göre, noktalı çizgi şehir bloğu uzaklığına göre, kesikli çizgi ise satranç tahtası uzaklığına göre orijinden $r$ uzaklıktaki tüm noktaların birleşimidir. Koordinat ekseninde orijin için $r$ uzaklıktaki tüm noktaların birleşimi, Öklid uzaklığına göre orijin merkezli ve $r$ yarıçaplı bir çember oluşturur. Şehir bloğu uzaklığına göre ise köşeleri eksenlerin $\mp r$ noktalarından geçen bir dörtgen oluşturur. Satranç tahtası uzaklığına göre bakılınca da kenarları eksenlerin $\mp r$ noktalarından geçen bir kare oluşturur [72]. Fiziksel uzayda Öklid uzaklık ölçütü en doğal mesafeyi verdiği için bu çalışmada Öklid fonksiyonu tercih edilmiştir [70].




Şekil 4.5. Koordinat ekseninde Öklid, şehir bloğu ve satranç tahtası ölçütüne göre orijinden *r* uzaklıktaki noktalar.

## 4.2 A* ALGORİTMASI

A* algoritması, Peter Hart, Nils Nilsson ve Bertram Raphael tarafından önerilen bir yol bulma algoritmasıdır [73]. Dijkstra algoritmasının özel bir uzantısı olan bu algoritma, aramayı belirli bir başlangıçtan bitişe odaklamak için ek olarak bitiş koordinatlarını kullanmaktadır [68]. Arama genişletildikçe, çevreleyen tüm düğümler üzerlerinden geçen en iyi yolun bir tahminine göre değerlendirilip sıralanarak bir sonraki düğüm elde edilir [74]. Sezgisel yöntem sayesinde A* algoritması bir başlangıçtan bir bitişe, Dijkstra algoritmasına kıyasla daha az sayıda düğüm genişleterek daha verimli bir şekilde yol üretebilmektedir [75].

A* algoritmasında bir düğümün kalitesi, üretim maliyeti ve sezgisel maliyet fonksiyonlarının toplanmasıyla ölçülür. Mevcut durumda genişletilmekte olan bir $n$ düğümünün toplam maliyet fonksiyonu $f(n)$, Eşitlik 4.4 ile gösterilebilir [69].

$$f(n) = g(n) + h(n) \tag{4.4}$$

Bir $n$ düğümünün üretim (generation) maliyeti $g(n)$ ile gösterilir ve başlangıç (start) $s$ düğümünden geçerli $n$ düğümüne ulaşmanın biriken gerçek mesafesini ölçer [76]. Bu değer Eşitlik 4.5 kullanılarak hesaplanabilir.

$$g(n) = \begin{cases} 0 & n = s \\ g(p(n)) + d(p(n), n) & n \neq s \end{cases} \tag{4.5}$$

Ebeveyn (parent) düğüm fonksiyonu $p(n)$, geçerli $n$ düğümüne geçilmeden önce işleme alınmış $n$ düğümünün komşusu olan düğümdür. Mesafe (distance) fonksiyonu $d(a,b)$ ise, $a$ düğümü ile $b$ düğümü arasındaki mesafenin ölçüsüdür. Mesafe ölçülürken Öklid, şehir bloğu veya satranç tahtası gibi uzaklık ölçütleri kullanılabilmektedir.

Sezgisel (heuristic) maliyet ise $h(n)$ ile gösterilir ve geçerli $n$ düğümü ile bitiş (target) $t$ düğümü arasındaki en kısa yol mesafesinin tahmini bir ölçüsüdür. Bulunan değer, bitişe kalan gerçek yol uzunluğuna her zaman eşit olmadığı için sezgisel bir değerdir. Sezgisel maliyet Eşitlik 4.6 ile gösterilebilir.

$$h(n) = d(n,t) \tag{4.6}$$

Toplam maliyet fonksiyonu, sezgisel maliyetin değerine göre farklı davranışlar göstermektedir. Teorik olarak sezgisel maliyet yeterince iyi geliştirilirse, sonuç daha kısa sürede elde edilebilir [77].

- Eğer sezgisel maliyet her durumda sıfır ise $(h(n) = 0)$ A* algoritması, bilgisayar bilimcisi Edsger W. Dijkstra tarafından önerilen Dijkstra algoritmasına dönüşür [78].
- Eğer sezgisel maliyet değeri en kısa yol tahmininden küçük ise $(h(n) < d(n,t))$ A* algoritması, daha çok düğüm genişleterek yavaş çalışacaktır.
- Eğer sezgisel maliyet değeri en kısa yol tahmininden büyük ise $(h(n) > d(n,t))$ A* algoritması, daha az düğüm genişleterek hızlı çalışacaktır. Ancak bu durumda elde edilen yol, en kısa yol olmayabilir.

A* algoritmasını açıklamadan önce bir düğümün açık ve kapalı olma durumu tanımlanmalıdır. Oluşturulan bir düğüm, toplam maliyeti hesaplanmasına rağmen henüz genişletilmemişse açık olarak adlandırılır. Eğer bu düğüm, vâris oluşturmak için genişletildiyse de kapalı olarak adlandırılır [69].

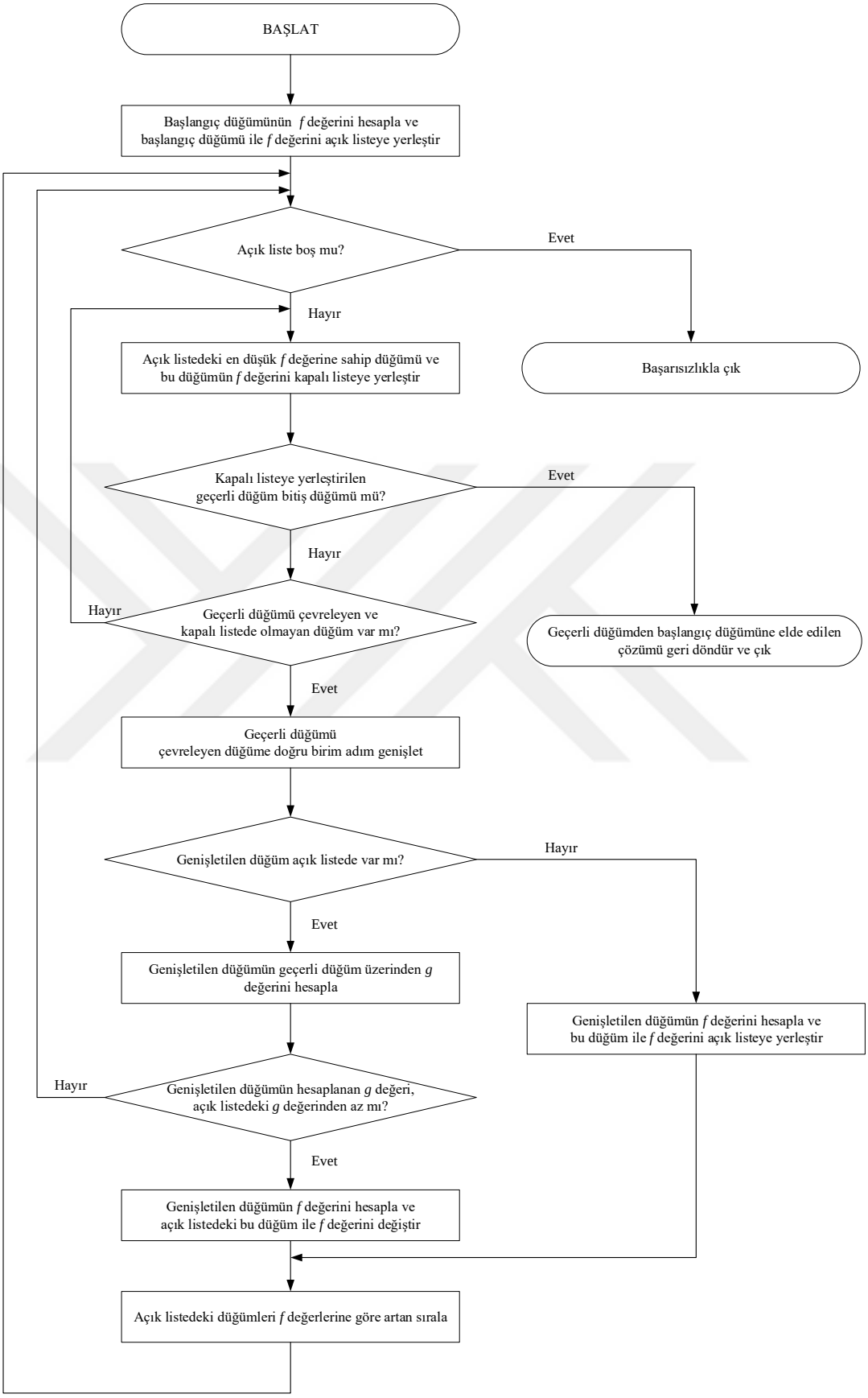
Oluşturulan tüm düğümlerin bilgileri, açık liste ve kapalı listede depolanır [74]. Genel olarak kontrol edilmesi gereken vâris düğümlerin bilgilerinin tutulduğu listeye açık liste denir. Hâlihazırda genişleterek keşfedilmiş ve planlanan yolda kullanılabilecek ebeveyn düğümlerin tutulduğu listeye ise kapalı liste denir [76]. Kapalı düğümler, bulunan en iyi yolun geçtiği düğümler belirlenirken birden çok kez işlenebileceğinden dolayı kolay takip edilebilmeleri için kapalı listeye eklenmektedir.

Listelere kaydedilen her düğüm kimlik $(\mathrm{ID})$, ebeveyn düğüm $(p(n))$, üretim maliyeti $(g(n))$, sezgisel maliyet $(h(n))$ ve toplam maliyet $(f(n))$ nitelikleriyle karakterize edilir [76].

Arama, başlangıç $s$ düğümü için komşu düğümlerin genişletilmesiyle başlar. En düşük toplam maliyet $(f)$ değerine sahip komşu düğüm, genişletilecek düğümlerin öncelik sırasına göre depolandığı açık listeden seçilir, genişletilir ve kapalı listeye eklenir. Her yinelemede bu işlem tekrarlanır. Geçerli düğüm için komşu düğümler keşfedilirken bazı koşullar doğrulanmalıdır. Bunlar aşağıdaki gibidir [76]:

- Bir komşu düğüm, kapalı listede zaten varsa yok sayılır.
- Bir komşu düğüm, açık listede zaten varsa önceden hesaplanmış diğer düğüm üzerinden ve geçerli düğüm üzerinden bu komşu düğüme ulaşmanın üretim maliyetleri $(g)$ karşılaştırılır. Eğer geçerli düğümü kullanmanın maliyeti daha düşük ise, komşu düğümün ebeveyn düğümü geçerli düğüm olacak şekilde değiştirilerek maliyet fonksiyonları yeniden hesaplanır.

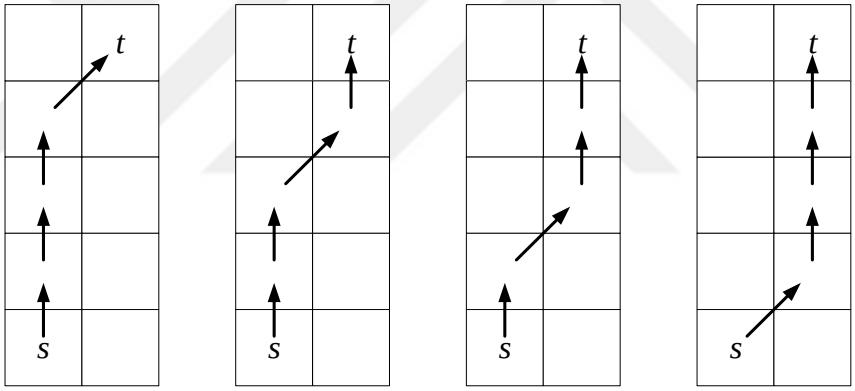
Bu süreç bitiş $t$ düğümüne ulaşılana kadar tekrar edilir. Bitiş düğüme ulaşıldığında ise, başlangıç düğüme ulaşana kadar geriye doğru (her düğümden ebeveyn düğüme geçilmesiyle) gidilerek ızgara haritasındaki en kısa yol bulunur [76]. A* algoritması için akış diyagramı Şekil 4.6'da gösterilmiştir.




Şekil 4.6. A* algoritmasının akış diyagramı.

## 4.3 ATLAMA NOKTASI ARAMA

Eşkenar hücreli ızgara tabanlı haritalarda, A* algoritmasının yol simetrisi ve düğüm genişlemesindeki artış olmak üzere iki zayıf yönü açığa çıkmaktadır. Bu iki zayıflık, A* algoritmasını çok yavaş kıldığı gibi, arama için büyük bir bellek alanına ihtiyaç duyulmasına ve gereksiz düğümler oluşturulmasına neden olmaktadır. Örnek olarak Şekil 4.7'de gösterilen 2 × 5 boyutunda ve engel içermeyen tek düzenli ızgara tabanlı bir grafik incelendiğinde, sol alt köşeden sağ üst köşeye kadar en kısa yol için dört özdeş maliyete sahip yol olduğu görülmektedir [79]. A* algoritması daha kısa bir yolun bulunamadığını doğrulamak için bu yollar boyunca aynı düğümleri fazladan kontrol etmektedir. Bu sorunların üstesinden gelinebilmesi için, belirli anahtar bilgilerin soyutlanmasıyla elde edilen harita temsillerine odaklanılarak arama alanı azaltılabilmektedir [80].

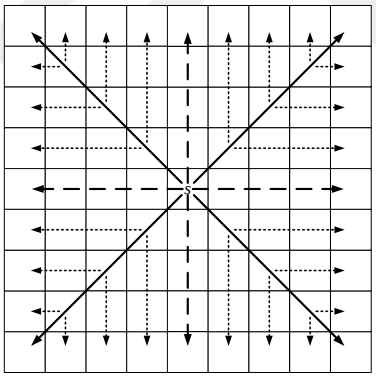



Şekil 4.7. Kare hücreli 2 × 5 boyutlu ızgaranın çapraz köşelerinde bulunan bir başlangıç düğümünden bir bitiş düğümüne eşdeğer yollar.

Daniel Harabor ve Alban Grastien tarafından önerilen atlama noktası araması, tek düzenli (uniform) ızgara haritalarında dikkate alınarak hesaplanması gereken birçok düğümün üzerinden atlayarak A* algoritmasının yol bulmasını hızlandıran bir simetri kırma iyileştirmesidir [81]. Atlama noktası aramasında, geçerli düğümün ebeveyninden her bir komşusuna uygun bir yolun (simetrik ya da başka türlü) var olduğu ve bu yolun geçerli düğümü ziyaret etmeyi içermediği incelenerek genişletilen toplam düğüm sayısı önemli ölçüde azaltılabilmektedir [82].

Atlama noktası araması, yol simetrisi problemini sistematik olarak çözmek için kurallı sıralama yöntemini kullanır ve arama alanını azaltmak için ızgara haritaları üzerinden atlama noktalarını tanımlar. Ardından, atlama noktaları arasından en uygun yol aranır. Ancak, atlama noktası aramasında bir atlama noktasından diğerine gerçekte atlanmaz. Bunun yerine, her bir ızgara hücresini kontrol etmek için atlama noktaları arasında yuvarlanılır [80].

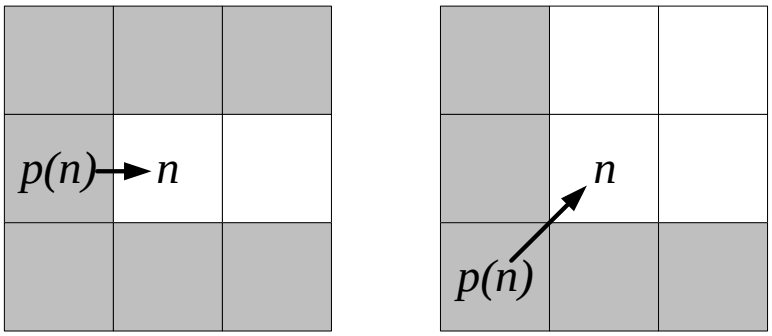
Şekil 4.8'de başlangıç $s$ düğümü genişletilirken yalnızca bir kez ziyaret edilen düğümlere ebeveynlerinden ulaşılma yönü gösterilmektedir. Düz yönlerdeki (yukarı, aşağı, sağa, sola) genişlemeler incelendiğinde, yalnızca aynı yönde karşılaşılan vâris düğümlerin olası komşu olarak değerlendirildiği görülür. Çapraz yönlerdeki genişlemeler incelendiğinde ise, önce ilgili yönün yatay ve dikey bileşenlerinin genişletildiği sonrasında aynı yönde karşılaşılan vâris düğümlerin olası komşu olarak değerlendirildiği görülür. Bu sayede atlama noktası araması, düğümlerin birden çok kez ziyaret edilmesini engeller [79].



Şekil 4.8. Atlama noktası aramasında çapraz yönde (düz çizgi) ilerlenirken işlenen yönler (noktalı çizgili) ve ilk önce işlenen düz yönler (kesikli çizgili).

Şekil 4.9'da gösterilen gri düğümler, değersiz komşulardır. Ok, geçerli $n$ düğümüne ebeveyninden gelinen düz ya da çapraz hareket yönünü göstermektedir. Gösterilen her iki durumda, geçerli $n$ düğümünden geçilmeden ebeveyni üzerinden en uygun şekilde ulaşılabilen tüm gri komşular budanabilir. Bu gri komşulara, geçerli düğümün ebeveyninden ulaşan ancak geçerli düğüm üzerinden geçmeyen daha az ya da eşdeğer maliyetli bir yol vardır. Beyaz düğümler, doğal komşulardır. Arama genişletilirken yalnızca doğal komşuların dikkate alınması yeterlidir. Atlama noktası araması, doğal

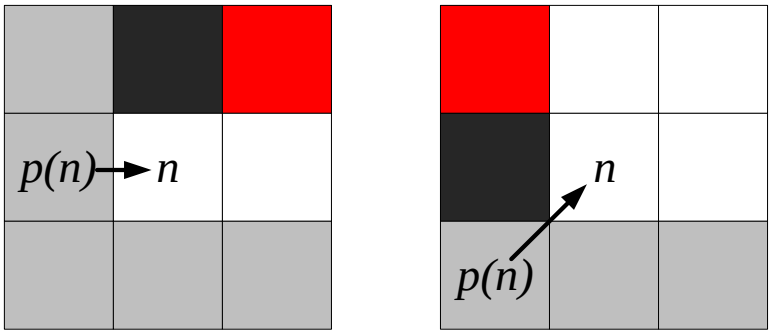
komşulara doğru genişlerken değersiz komşuları hesaplamaya dâhil etmez. Yinelemeli olarak uygulanan bu kural, ağaç dallarının budanmasına benzetilerek budama kuralı adını almıştır.




(a) (b)

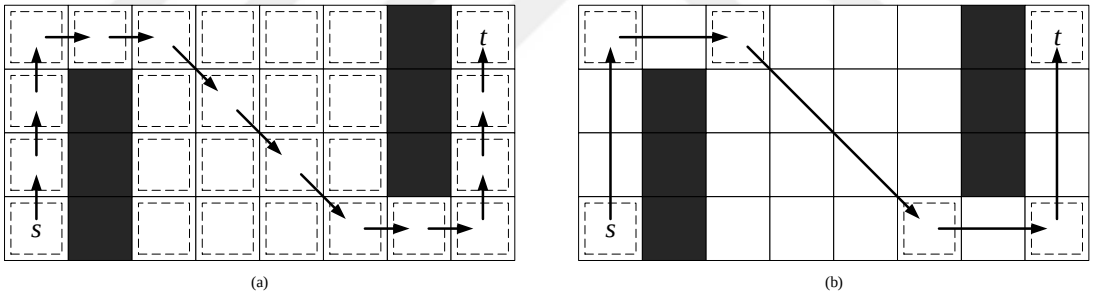
Şekil 4.9. Genişletilen geçerli *n* düğümü için komşularının budanması. a) Düz yönde budama. b) Çapraz yönde budama.

Şekil 4.10'da gösterilen ok, geçerli *n* düğümüne ebeveyninden gelinen düz ya da çapraz hareket yönünü göstermektedir. Geçerli *n* düğümünün kırmızı renkle vurgulanan komşusu, bir engele bitişik olan ilgili düğümün ebeveyninden ulaşılamadığı için budanamaz. Görünen siyah düğümler ise engeli göstermektedir. Engeller, üzerinden bir yolun geçemediği harita unsurlarının temsilîdir. Kırmızı düğümler, zorunlu komşulardır. Geçerli düğümün bitişiğindeki engel, zorunlu komşuya geçerli düğümü ziyaret etmeden daha az maliyetle ulaşılmasının önünü kesmektedir. Atlama noktası aramasında, uygulanan budama kuralının yinelenmesi zorunlu komşu tespit edildiğinde ya da genişleme yönünde bir engele çarpıldığında durdurulur. Zorunlu komşu tespit edildiğinde, geçerli düğüme atlama noktası vârisi adı verilir. Elde edilen atlama noktaları, zorunlu komşusuna ulaşan en uygun yolun üzerinde bulunmaktadır.




Şekil 4.10. Genişletilen geçerli *n* düğümü için zorunlu komşular. a) Düz yönde zorunlu komşu. b) Çapraz yönde zorunlu komşu.

Atlama noktası araması, açık listeye sadece atlama noktaları adı verilen düğümleri koymaktadır. Açık listede yer alan düğümlerdeki önemli bir azalma, A* algoritmasında etkili bir çalışma süresi tasarrufu sağlamaktadır. Bu duruma örnek olarak verilen Şekil 4.11'de, açık listeye yerleştirilen düğümler kesikli çizgili kare ile belirtilerek atlama noktası aramasının A* algoritmasıyla kıyası gösterilmiştir [79].




Şekil 4.11. A* algoritmasında (a) ve atlama noktası aramasında (b) oluşturulan düğümlerin kıyaslanması.

Atlama noktası araması en uygun yolu bulurken ön işlem gerektirmez ve belleğe aşırı yüklemede bulunmaz. Bulunacak en uygun yol ne kadar uzunsa, atlama noktası araması uygulamasından da o kadar fazla fayda elde edilir. Ayrıca atlama noktaları arasındaki arama hızının da artırılabilmesi için, ön işleme yoluyla atlama noktası haritası oluşturulabilir [79]. Bu sayede, bitişik atlama noktalarına ilişkin mesafe bilgisi atlama noktası haritasında mevcut olduğundan atlama noktaları arasındaki yol doğrudan bulunabilir [80].

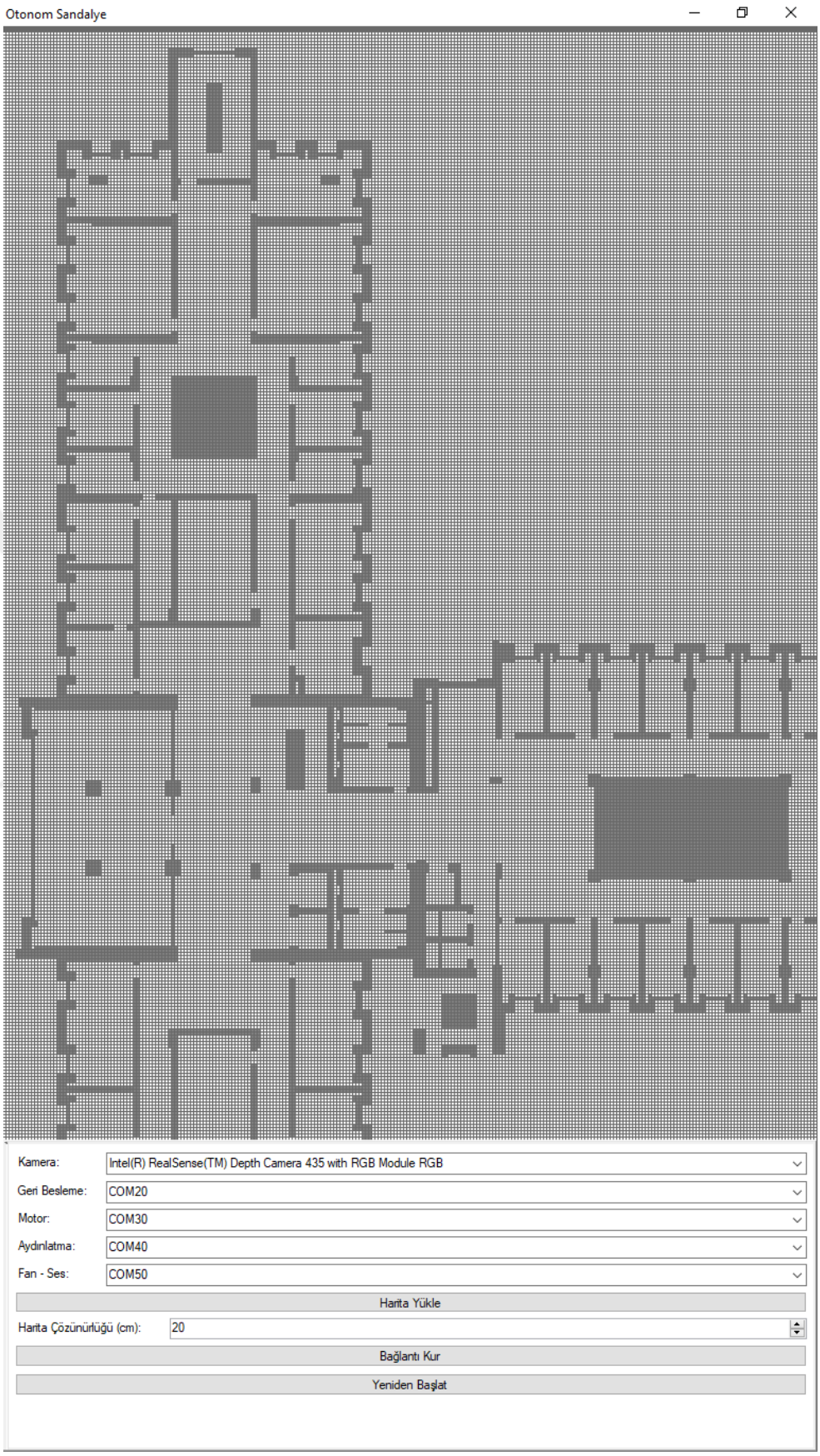
Bu çalışmada, en verimli algoritmalardan biri olan A* algoritmasının atlama noktası araması optimizasyonu kullanılmıştır. Bu kapsamda, Woong Gyu La tarafından C# dilinde yazılan ve açık erişimli olan EpPathFinding.cs kütüphanesinden yararlanılarak ızgara tabanlı harita grafiği üzerinde güzergâh belirlemesi yapılmıştır [83].

# BÖLÜM 5

# ARAYÜZ VE MODELLEME

Akıllı tekerlekli sandalye sistemine ait otonom kontrol yazılımının oluşturulması için Microsoft Visual Studio üzerinden C# dilinde görsel arayüz tasarlamıştır. Şekil 5.1'de, görüldüğü gibi tüm bilgi girişleri, tekerlekli sandalyede ekranın konumlandırılma durumuna göre kullanıcı elinin rahatlıkla ulaşabileceği en alt kısma yerleştirilmiştir. Arayüzün üst kısmı ise yüklenen harita temsilinin gösterilmesi için ayrılmıştır. Yazılımın ilk açılışında, ayarlar sayfası açılarak kullanıcıdan bazı parametreleri girerek doğrulaması istenir.

Akıllı tekerlekli sandalye sisteminde takılı bulunan kameralar, "Kamera" açılır menüsünde listelenmektedir. Burada seçilen kamera, bir sonraki güzergâh belirleme sayfasında yansıtılan kamera görüntüsünün kaynağını seçer. Kamera görüntüsü, seçilen kameradan elde edilen görüntüleri anlık olarak sunarak kullanıcıya ek bir görüş açısı sunmaktadır. "Geri Besleme" olarak istenen mikrodenetleyici, tekerlekli sandalyenin hız sensörü, acil durdurma butonu, gerilim ölçeri ve sıcaklık ölçeri gibi elemanlarından elde edilen bilgileri bilgisayara aktarmada kullanılmaktadır. "Motor" mikrodenetleyicisi, doğrudan çift DA motor sürücü kartına ve fren sistemlerini açıp kapatan rölelere bağlıdır. "Aydınlatma" mikrodenetleyicisi, tüm LED lambalarını açıp kapatan sinyalizasyon işlemlerinden sorumludur. "Fan – Ses" mikrodenetleyicisi ise, sistemin soğutulmasında kullanılan fanların hız ayarının yapılmasını ve vızlayıcı (buzzer) ile ses çıkarılarak çevredeki diğer insanların uyarılmasını sağlar. Ayarlar sayfasının en altında yer alan "Harita Yükle" butonu ile ikili (binary) değerde tek renkli bit eşlem resminin (bitmap, .bmp) yüklenmesi istenir. Harita temsilinde kullanılacak olan resimdeki her bir piksel için gerçek uzayda karşıladığı karesel bölgenin bir kenar uzunluğu santimetre cinsinden "Harita Çözünürlüğü" sayısal değer alanına girilmektedir.



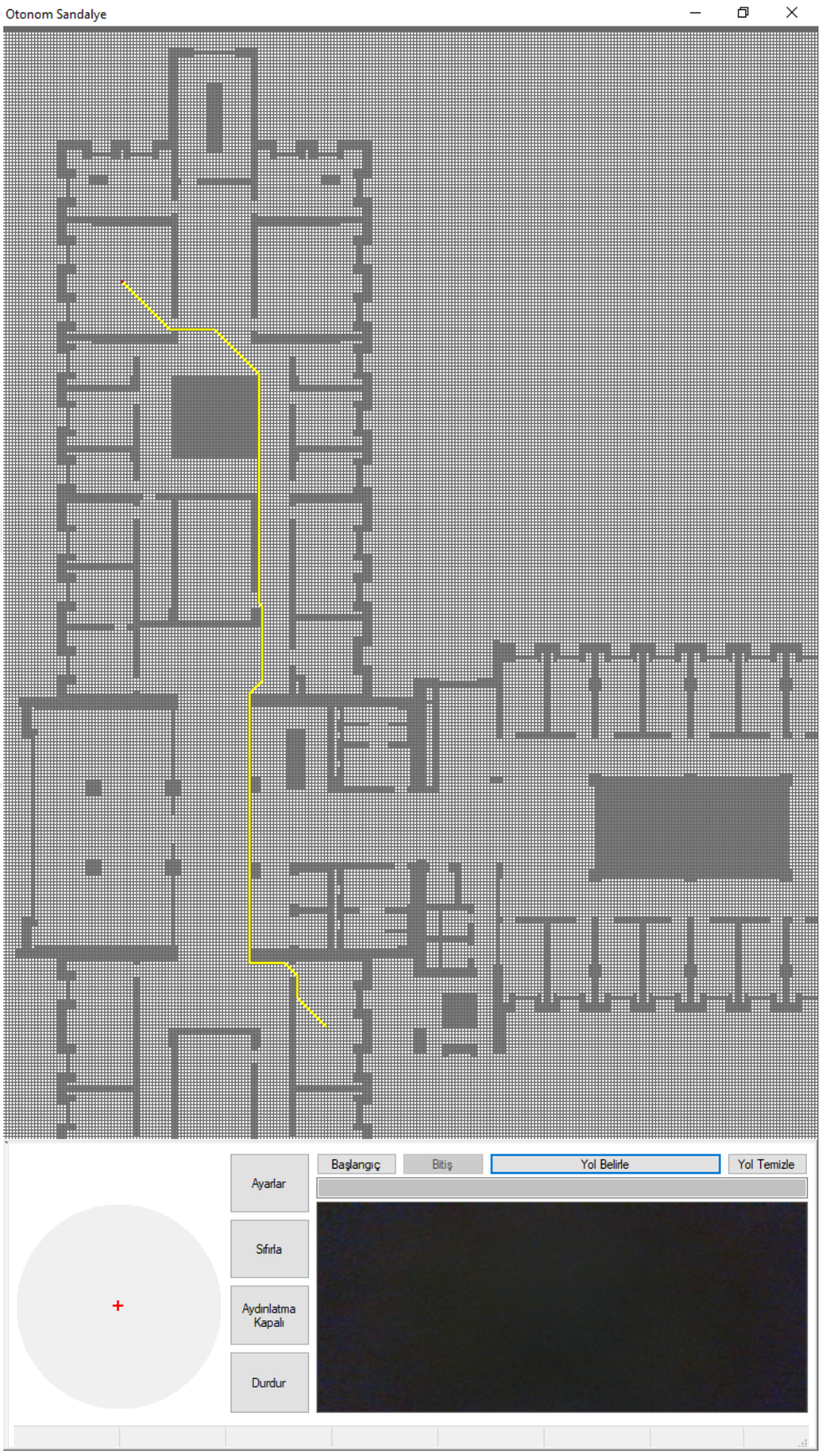

Şekil 5.1. Arayüzde ayarların girilmesi.

Şekil 5.2'de, görüldüğü gibi arayüzün üst kısmında harita temsilî yansıtılmaktadır. Bu temsil üzerinde güzergâhın belirlenebilmesi için sağ altına dört buton yerleştirilmiştir. Bu butonlardan "Başlangıç" bulunulan konumun işaretlenmesinde, "Bitiş" gidilmek istenen konumun işaretlenmesinde, "Yol Belirle" bulunulan konum ile gidilmek istenen konum arasındaki en kısa yolun bulunmasında, "Yol Temizle" ise belirlenen güzergâhın silinmesinde kullanılmaktadır.

Görsel arayüzde yer alan diğer butonlar ise çeşitli işlevlerin sağlanabilmesi için arayüze eklenmiştir. Bu butonlardan "Ayarlar", akıllı tekerlekli sandalye sistemine bağlı unsurlar için bilgisayarın seri haberleşme bağlantılarının belirlenmesi ve ölçekli harita görselinin yüklenmesi için kullanılmaktadır. "Sıfırla" butonu ise, arayüz kodunda yer alan tüm değişkenleri varsayılan değerlerine geri döndürmektedir. Akıllı tekerlekli sandalyede yer alan ön ve arka aydınlatmaların açılabilmesi için de "Aydınlatma Kapalı/Açık1/Açık2/Açık3" butonu konulmuştur. Bu buton üzerine her tıklamada, ön aydınlatmada kullanılan LED lambalar kademeli olarak açılmaktadır. En altında yer alan "Durdur" butonu ise tekerlekli sandalyede yer alan acil durdurma butonunun yazılımsal işlevinin tetiklenmesini sağlamaktadır. Bu butona tıklatıldığında sürdürülen işlemler durdurularak tekerlekli sandalyenin hareketine devam etmesi engellenmektedir.

Görsel arayüze manuel kontrol için bir yönetme kolu (joystick) bölgesi de eklenmiştir. Bu bölge, görsel arayüzün sol alt tarafında görünen merkezi kırmızı artı ile işaretlenmiş bir daire şeklindedir. Daire üzerinde dokunulan herhangi bir nokta için, merkeze göre hareket edilmek istenen yön ve merkezden uzaklığa göre de hareket hızının büyüklüğü belirlenebilmektedir.

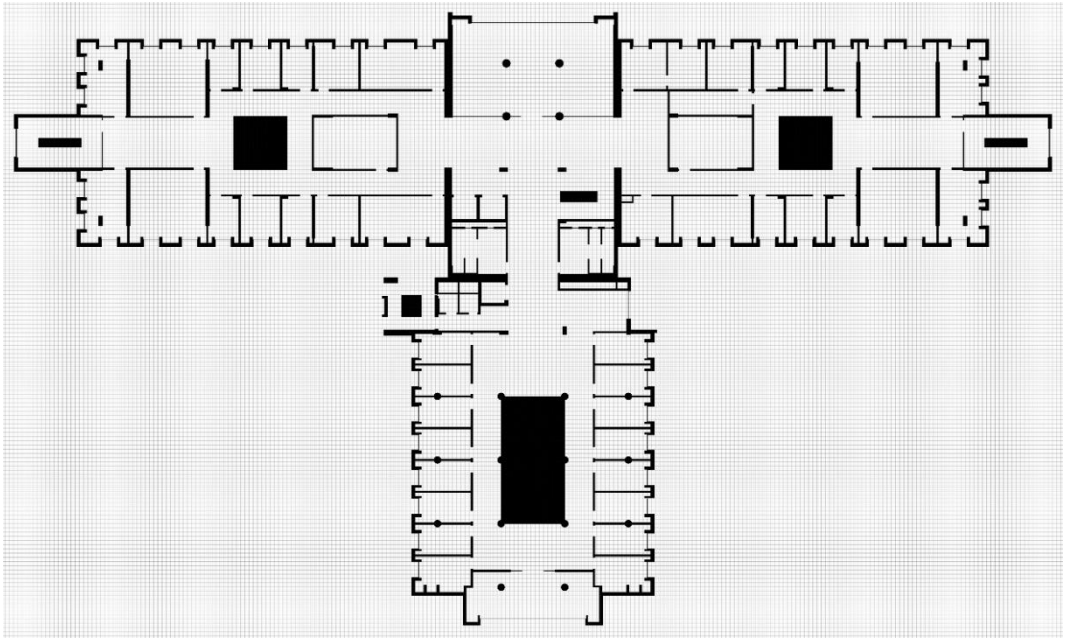
Görsel arayüzün sağ alt kısmında geriye kalan siyah renkli dikdörtgen bölge ise ayarlar sayfasında seçilen kameradan alınan anlık görüntülerin yansıtılması için konulmuştur. Bu alan, akıllı tekerlekli sandalye sistemine takılan derinlik algılayıcı kameranın görüş alanının açık olup olmadığının denetlenmesinde kullanılır.



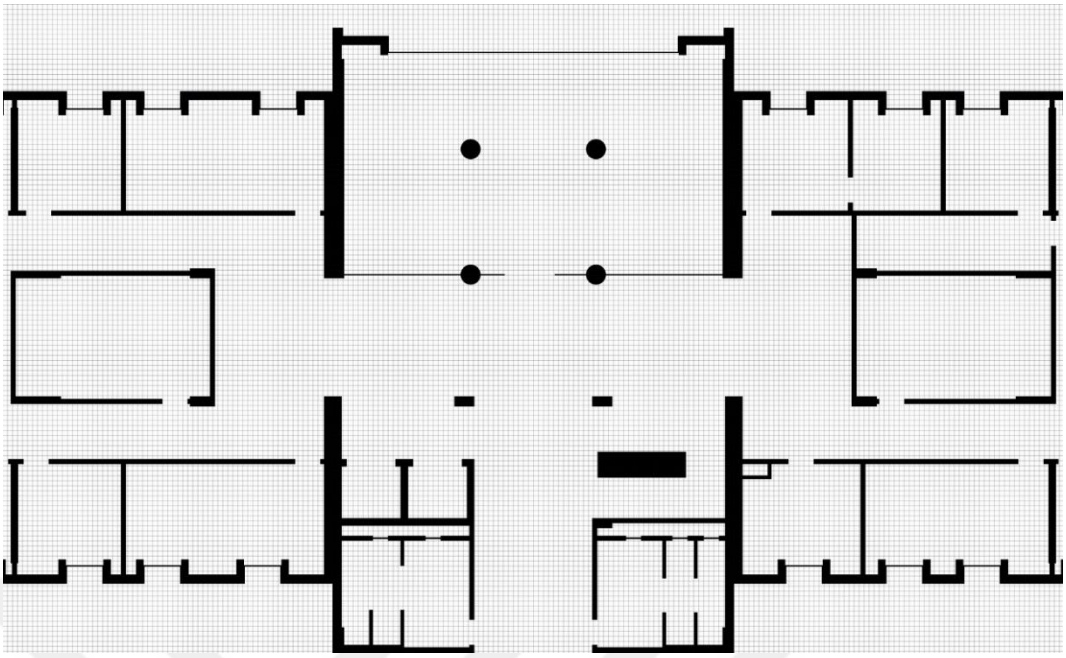

Şekil 5.2. Arayüzde güzergâh belirlemesi örneği.

Yol bulma işlemi öncesinde, başlangıç ve bitiş konumunun belirtilmesi gerekmektedir. Ardından atlama noktası araması kullanılarak uygun bir en kısa yol belirlenir. Algoritma kapsamında çapraz olarak ilerlenebilmesi için, yolun geçtiği temsilî haritaya ait ızgara hücrelerinin köşeleri üzerinden geçilebilir olması gerekmektedir. Yolun yanında üzerinden geçilemez bir ızgara hücresi varsa, çapraz olarak ilerleme yerine köşeden dolanma tercih edilmektedir.

Bilgisayar destekli tasarım yazılımlarında hazırlanan iki boyutlu ortam çizimleri, tek renkli bit eşlem resmi olarak kaydedilebilir. Bu sayede bir ortam çizimi, ölçekli pikselleme (pixelation) işleminden geçirilerek .bmp dosya türünde dışa aktarılır. Akıllı tekerlekli sandalye sisteminin denendiği ortama ait kat planı çizimi, benzer şekilde harita görüntüsüne dönüştürülmüş ve duvar ile pencere gibi engellerin siyah renkli piksel kümesi olarak belirtilmesi sağlanmıştır. Elde edilen temsilî haritalama görüntüsü arayüze aktarıldıktan sonra, tekerlekli sandalyenin bulunduğu konum ve gidilmek istenen konum seçilebilmektedir. Ortamın önceden haritalanması ve uygun yol bulma algoritmalarının kullanılmasıyla, kullanıcının ulaşmak istediği bir konuma gidebileceği en kısa yol planlanabilmektedir. Şekil 5.3 ve Şekil 5.4, Karabük Üniversitesi Mühendislik Fakültesi binasının dördüncü kat planını, her hücresi 20 cm × 20 cm çözünürlüğüne sahip bir ızgara üzerinde göstermiştir.

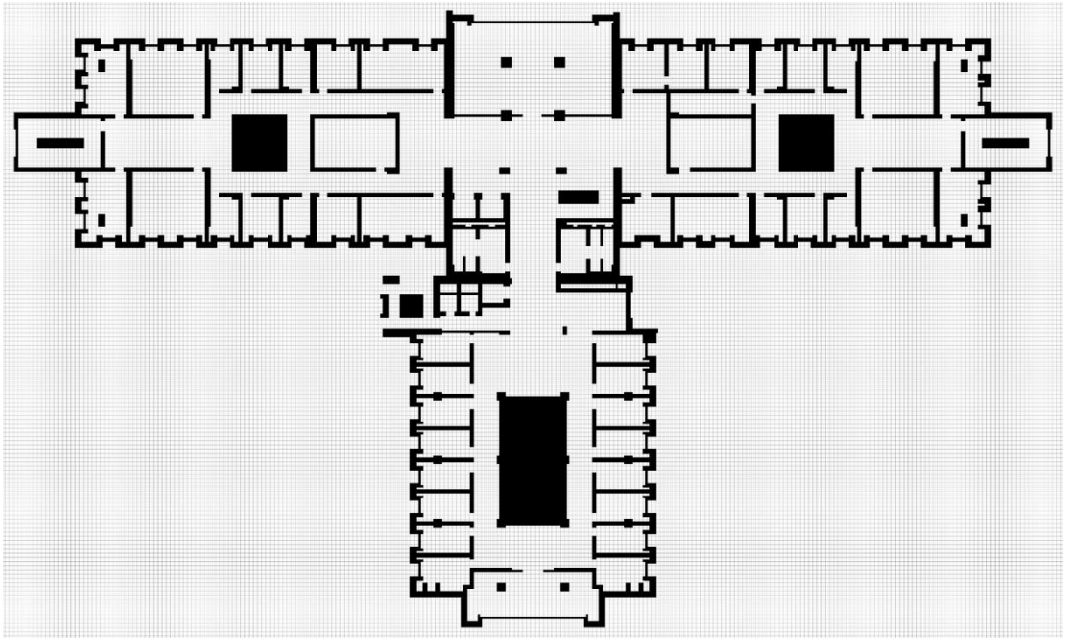


Şekil 5.3. Kat planının 20 cm × 20 cm hücreli ızgarada temsilî.

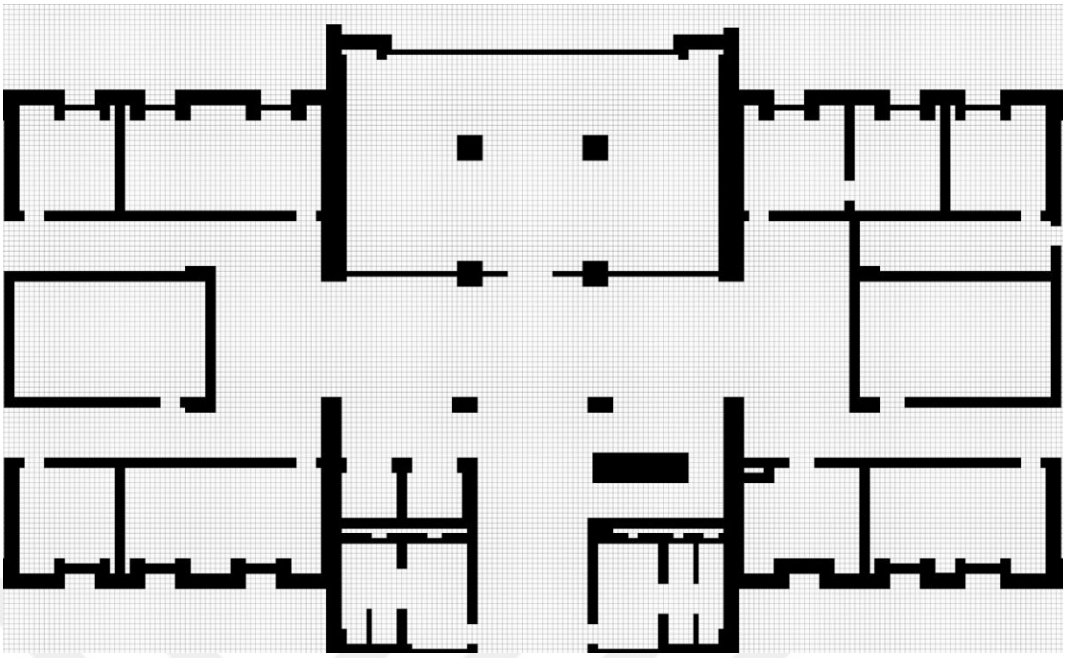


Şekil 5.4. Kat planının 20 cm × 20 cm hücreli ızgarada temsilî (yakın görünüm).

Kat planındaki ızgara hücreleri arasında, hücreyi tam olarak dolduramayan engeller de bulunmaktadır. Bu kısmi engele sahip ızgara hücreleri, yol güvenliğinin arttırılması için Şekil 5.5 ve Şekil 5.6 üzerinde görüldüğü gibi tam dolu ızgara hücrelerine dönüştürülmüştür. Yapılan pikselleştirme (pixelization) işlemi sonucunda elde edilen harita temsilinde tüm yassı köşeler düzleştirilmiştir.



Şekil 5.5. Izgara üzerine oturtulmuş kat planının pikselize edilmesi.



Şekil 5.6. Izgara üzerine oturtulmuş kat planının pikselize edilmesi (yakın görünüm).

Belirlenen güzergâh boyunca yapılan hareket esnasında, harita temsilî üzerinde gösterilmeyen dinamik ya da statik engeller ile karşılaşılabilir. Bu engellerin tespit edilebilmesi için derinlik algılayıcı kameradan gelen bilgiler kullanılarak arayüz yazılımı içerisinde değerlendirilmektedir. Derinlik algılayıcı kameranın sunduğu standart kamera görüntüsü, makine görüşünün kontrol edebilmesi için Şekil 5.2'de gösterilen arayüzün sağ alt kenarında yansıtılmaktadır. Elde edilen standart kamera görüntüleri işlenerek çevredeki QR kodlarının değerlendirilmesi de yapılmaktadır. Bunun için görüntüdeki QR, PDF417 (Portable Data File 417, Taşınabilir Veri Dosyası 417), EAN (European Article Number, Avrupa Madde Numarası), UPC (Universal Product Code, Evrensel Ürün Kodu) ve Karekod gibi barkod kodlarının çözülmesini destekleyen açık erişimli ZXing.Net kitaplığı kullanılmıştır [84].

QR kodlar ile akıllı tekerlekli sandalye sisteminin bulunduğu ortamdaki oda ve benzeri özel yerler etiketlenebilir. Gerekli görüldüğünde asılan QR kodlarının sayısı arttırılarak sistemin bilgilendirilmesi için ayrıntılı işaretçiler yerleştirilebilir. Tüm işaretçiler, güzergâh belirleyerek otonom kontrol sağlayan yardımcı yazılım üzerinden değerlendirilmektedir.

Akıllı tekerlekli sandalye sistemi, özellikle kalabalık ortamlarda hareket hâlindeyken yaşanabilecek çarpmalara karşı engel tespiti yapmaktadır. Derinlik algılayıcı kameranın, nokta bulutu verisi sürekli değerlendirilerek yakında bir engel olup olmadığı kontrol edilmektedir. Aniden bir engel çıkması durumunda ise, sesli ikaz verilerek alternatif güzergâh üzerinden yola devam edilmesi için kullanıcıya bilgi verilmektedir. Kablosuz kulaklık sayesinde, ikaz ve bilgilerin sesli olarak doğrudan kullanıcıya iletilmesi sağlanmaktadır.

Derinlik algılayıcı kameradan alınan engel bilgisi doğrultusunda, motor sürücü kartına uygun hareket komutları gönderilerek güzergâha göre otonom kontrol sağlanmaktadır. Bu aşamada, bulunan en kısa yolun her bir düğümünden geçilerek istenilen konuma ulaşılabilmesi için bir kontrol modeli geliştirilmelidir. Geçerli düğümden sıradaki düğüme ilerlemede kullanılacak kontrol modelinin akıllı tekerlekli sandalye sisteminin her bir motoru için hız ve yönü belirlemesi gerekir. Ayrıca konumlandırma hatalarını en aza indiren bir denetleyici de tasarlanmalıdır [85].

Doğrusal ve doğrusal olmayan iki tür denetleyici vardır. Doğrusal denetleyiciler, bir dizi denklemin analizi sonucunda hassas veya yaklaşık olarak modellenebilen sistemlerde kullanılır. Doğrusal olmayan denetleyiciler ise karmaşık davranışı nedeniyle matematiksel model belirsizliği içeren sistemlerde kullanılır. Yapısı gereği tüm fiziksel sistemler doğrusal değildir [86].

Matematiksel modelleme mümkün olduğunda, kontrol teorisi daha güvenli bir yaklaşım sunar. Fakat, pratik problemlerin çözümünde çok fazla çalışma gerektirebilir. Modelleme zor veya maliyetli olduğundan bulanık mantık, bir denetleyici tasarlamak için makul, kullanışlı ve şeffaf bir araç sunmaktadır [87]. Akıllı tekerlekli sandalye sistemi için kullanıcı bir gezinme kararı verdiğinde bulunması gereken matematiksel model de oldukça karmaşık olmaktadır. Bu nedenle yapılması gereken hareketlerin tahmin edilmesi, bulanık mantığa dayalı bir denetleyici ile mümkündür [88].

Bu çalışmada, hareket yönüne göre tekerlek hızlarının uygun olarak ayarlanabilmesi için bulanık mantık denetleyicisi kullanılmıştır. Akıllı tekerlekli sandalyenin bulunduğu düğüm ile güzergâhındaki en yakın düğüm arasındaki mesafe ve yönelme

açısı kinematik modellemeden çıkarılmaktadır. Denetleyici, tekerlekli sandalyenin mevcut hızına göre bir sonraki hareketi için yeni teker hızlarını belirlemektedir. Geliştirilen bulanık mantık denetleyicisi hareket yönlendirmesinin yanı sıra engellerden kaçınmada da görev almaktadır.

## 5.1 BULANIK MANTIK

Mantık, mevcut önermelerden yeni önermeler elde etmek anlamına gelen muhakeme yöntem ve ilkelerinin incelenmesidir. Klasik mantıkta önermelerin doğru ya da yanlış olması gerekir. Bir önermenin doğruluk değeri 0 veya 1 olabilir. Bulanık mantık, bir önermenin doğruluk değerlerinin [0, 1] aralığındaki herhangi bir sayı olmasına izin vererek iki değerli klasik mantığı genelleştirir. Bu genelleme ile kesin olmayan önermelerden kesin olmayan sonuçlar, yaklaşık muhakeme olarak da adlandırılan bulanık muhakeme sonucunda çıkarılır. Lotfi A. Zadeh tarafından tanıtılan bulanık mantık, matematiksel modelin bilinmediği sistemler için bir kontrol tekniği olarak kullanılmaktadır [89].

Bulanık muhakemede temel öneri temsil birimi olarak dilsel değişkenler kullanılır. Dilsel değişkenler, değer olarak kelimeleri almaktadır. Günlük hayatımızda, bir değişkeni tanımlamak için genellikle birden fazla kelime kullanırız. Örneğin bir arabanın hızını dilsel bir değişken olarak görürsek, değerleri “yavaş değil”, “çok yavaş”, “biraz hızlı” ve benzeri olabilir. Dil değişkenleri de genel olarak birden çok terimin birleşmesinden oluşur. Bu terimler, bulanık kümelerin etiketleri olan birincil terimler (“yavaş”, “orta”, “hızlı” gibi), kısıtlamalar (“çok”, “hafif”, “az ya da çok” gibi) ve tamamlayıcı (“değil”) olarak gruplandırılabilir [90].

Bir bulanık kural, koşulu tanımlayan öncül (antecedent) ve koşulun geçerli olması durumunda çıkarılabilecek sonucu açıklayan ardıl (consequent) kısımlarından oluşur. Bu öncül ve ardıl kısımlar, dilsel değişkenler kullanılarak oluşturulmuş önermelerdir. Bir bulanık kural, “EĞER öncül, İSE ardıl.” olarak ifade edilebilir [90].

Bir bulanık kuralın öncülü, bir terimden veya bağlaç (“ve”, “veya”) ile birleştirilmiş birden fazla terimden oluşabilir [91]. Bulanık kurallara göre oluşturulan sistemlerin

bileşenleri arasındaki ilişki, bulanık mantıksal işlemlere göre kurulur. Bulanık mantıktaki çeşitli işlemlerin, klasik mantıkta bir karşılığı vardır [91]. Temel bulanık işlemleri ve bunların klasik mantıktaki karşılıkları Çizelge 5.1'de verilmiştir [92].

Çizelge 5.1. Bulanık mantıktaki temel işleçler.

| Sözel | Mantık | | Küme | | Bulanık |
|---|---|---|---|---|---|
| $A$ değil | olumsuzluk (negation) | $\neg A$ veya $\overline{A}$ | $A^c$ | tamamlayıcı (complement) | $1-a$ |
| $A$ ve $B$ | bağlanma (conjunction) | $A \wedge B$ | $A \cap B$ | kesişim (intersection) | $\min(a,b)$ |
| $A$ veya $B$ | ayrılma (disjunction) | $A \vee B$ | $A \cup B$ | birleşim (union) | $\max(a,b)$ |
| Eğer $A$ ise $B$ | ima (implication) | $A \rightarrow B$ | $A \subseteq B$ | alt ve kendisi (subsethood) | $\begin{cases} 1 & b \geq a \\ b & a < b \end{cases}$ |

## 5.2 BULANIK MANTIK DENETLEYİCİSİ

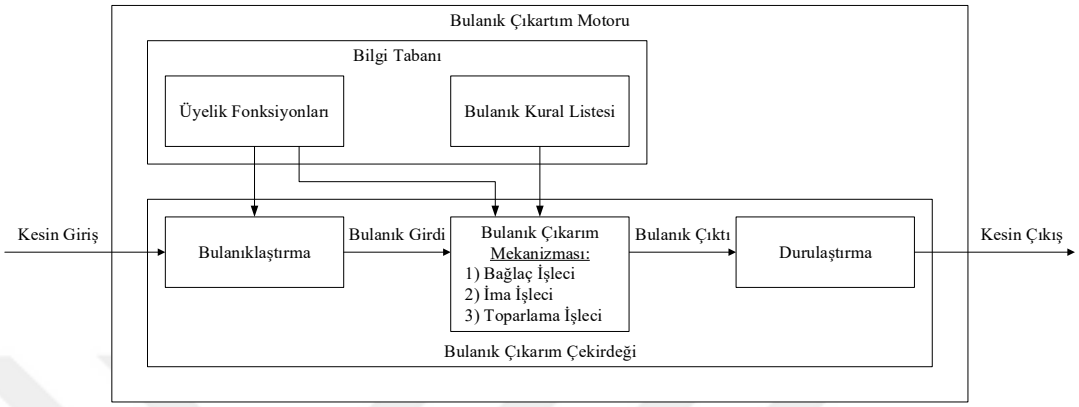
Bulanık küme teorisine dayanan bulanık sistemler, ilk kez Ebrahim H. Mamdani tarafından uygulanmıştır [93]. Bulanık sistemler, kontrol sistemleri, veri madenciliği, uzman sistemler ve örüntü tanıma gibi birçok uygulamada başarıyla kullanılmaktadır [94]. Bulanık sistemler, doğrusal olmayan süreçleri modellemeye ve kontrol etmeye izin verir.

Bir bulanık sistemin modellenmesinde, bulanık mantık hesaba katılarak sistem davranışını tanımlayan kurallar ve bulanık kümeler kullanılır [95]. Bulanık kontrol sistemlerinin tasarımı için toplanan bilgileri sunma yeteneklerine göre farklılık gösteren Mamdani ve Sugeno modelleri olmak üzere iki ana yöntem vardır [96].

### 5.2.1 Mamdani Modeli

Mamdani modeli kullanılarak oluşturulmuş bir bulanık mantık denetleyicisinin üç temel aşaması vardır. İlk aşama, kesin değer girişini bulanık denetim terimleri için karşılanma derecelerine dönüştüren bulanıklaştırmadır. İkinci aşama, kural listesini işleyip bulanık muhakeme sürecini uygulayan ve elde edilen sonuçları yerine getirme güçlülük derecesine göre birleştiren bulanık çıkarımdır. Bulanık mantığın üçüncü adımı ise, çıkarım motoru tarafından hesaplanan bulanık alt kümesini kesin değer

girişine dönüştüren durulaştırmadır. Bir bulanık mantık denetleyicisi (fuzzy logic controller, FLC) için Mamdani modeline göre sistemin temel yapısı Şekil 5.7'de gösterilmektedir.




Şekil 5.7. Mamdani modeline göre bulanık mantık denetleyicisi.

Bulanık mantık denetleyicisinin bulanıklaştırma ve bulanık çıkarım mekanizması süreçleri bir bilgi tabanına ihtiyaç duyar. Bilgi tabanındaki üyelik fonksiyonları girişin ve çıkışın bulanık kümelerini tanımlarken, bulanık kural listesi girdi ve çıktı bulanık kümeleri arasındaki ilişkileri tanımlamaktadır [94]. Mamdani bulanık sistemlerinde giriş ve çıkış üyelik fonksiyonları dilsel etiketlerle (yakın veya uzak gibi) açıklanır. Bulanık kural listesi de EĞER–İSE dilsel yapısı ile oluşturulur [88]. Mamdani bulanık sistemlerinde kuralların hem öncül hem de ardıl kısımları üyelik fonksiyonları ile temsil edilen dilsel terimlerden oluşmaktadır [93].

Bulanıklaştırma birimi, kesin giriş değerlerine karşılık gelen giriş üyelik fonksiyonlarını ve üyelik (eşleşme) derecesini bulur. Bu birimin çıktısı bulanık kümelerdir. Bulanık çıkarım mekanizması birimi ise bulanık girdi kümelerini her kurala göre değerlendirerek bir bulanık çıktı kümesi elde eder. İlk adım, incelenen kural için eğer öncül kısmında bulanık bağlaç varsa, ilgili girdi bulanık kümelerinin bu bağlaca göre birleştirilmesidir. Ardından, incelenen kuralın imasına göre ardıla karşılık gelen çıkış üyelik fonksiyonları ve ateşleme (etkinleştirme) gücü belirlenir. Son adımda ise her kuralın bulanık kümeleri toparlanarak bir bileşke bulanık küme içerisinde bir araya getirilir. Bulanık çıkarım mekanizmasından sonra gelen

durulaştırma birimi, çıkarım aşamasından elde edilen bulanık kümeden kesin çıkış değerini bulur.

Belirli bir kuralın öncülü birden fazla parçaya sahip olduğunda, tüm öncülü temsil eden bir doğruluk değerini elde etmek için ilgili bulanık bağlaç (“ve”, “veya”) işleci uygulanmalıdır. Bulanık bağlaç işlecini uygulamanın çeşitli yöntemleri vardır. Bunlar arasında en yaygın olanı, en düşük olanı alma (min) “ve” yöntemi ile en büyük olanı alma (max) “veya” yöntemidir [91].

Bulanık kurallar, uzman bilgisi ve deneyimi aracılığıyla tasarım değişkenlerini ayarlamak için kullanılır. Sezgisel olan bu kurallar, gerektiğinde kolaylıkla değiştirilebilirler. Bulanık çıkarım mekanizması her bir kuralın ardılını, öncülünün doyma derecesi ile ilişkili bir ima işleci uygulayarak yeniden şekillendirir. En çok kullanılan ima yöntemlerine en düşük olanı alma (min) ve matris çarpımı (product) örnek verilebilir [94].

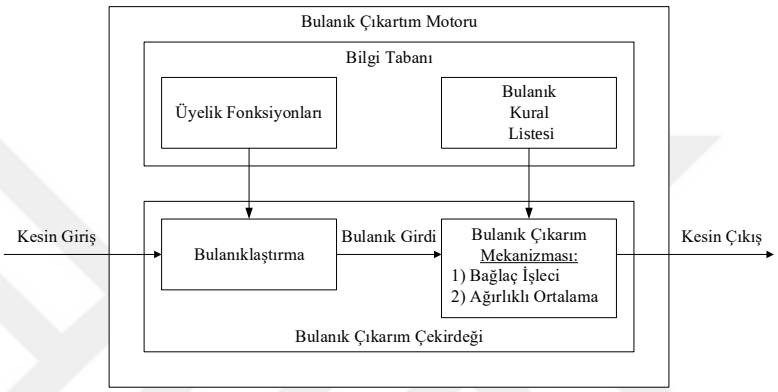
Bulanık çıkarım mekanizması aracılığıyla karar verilirken bilgi tabanındaki her kuralın katkısının hesaba katılması için kural değerlendirmesinde elde edilen bağımsız bulanık kümelerin toparlanması gerekir. Toparlama işleci için kullanılabilir çeşitli yöntemler vardır. Bunlar arasında, tüm girdiler arasından en büyük olanı alma (max) ya da tüm girdilerin cebirsel toplamını alma (sum) yaygın olarak kullanılan yöntemlerdir [91].

Bulanık çıkarım mekanizmasından elde edilen bileşke bulanık küme birden fazla kesin çıkış değerini kapsadığından, fiziksel bir sistemin denetlenmesinde uygun değildir. Bulanık çıktı kümesinin belirli bir sayısal değere dönüştürülmesi için durulaştırma gerekir. Durulaştırma işleminde ağırlık merkezi (centroid), toplamların merkezi (center of sums), en büyük üyelik dereceli elemanların orta noktası (mean of maxima) gibi birçok kullanılabilir yöntem vardır [91].

### 5.2.2 Sugeno Modeli

Tomohiro Takagi ve Michio Sugeno, Mamdani modeli kullanan bulanık sistemlerin davranışını belirleyen mekanizmayı bir adım daha ileri götürerek, bulanık sonuçların

kullanılmasının zorunlu olmadığını fark etmiş ve Sugeno bulanık modeli olarak bilinen yeni bir bulanık model türü önermiştir [97]. Bu model, Takagi-Sugeno (T-S) bulanık modeli olarak da adlandırılır [87]. Michio Sugeno ve Geun-Taek Kang ise Sugeno yöntemini genişletmiştir [98,99]. Bu nedenle, bazen Takagi-Sugeno-Kang (TSK) modeli olarak da anılmaktadır [88]. Sugeno modeli, karmaşık doğrusal olmayan sistemlerin dinamik olarak modellemesi için uygundur [90]. Şekil 5.8, bir Sugeno bulanık sistemini tanımlayan unsurları göstermektedir.




Şekil 5.8. Sugeno modeline göre bulanık mantık denetleyicisi.

Sugeno modelinde ilk olarak, bulanıklaştırma ile her kural için bir girdi değeri bir üyelik seviyesine dönüştürülür. Varsa eğer, çok parçalı öncül içeren her bir kuralın doğruluk değerinin elde edilmesi için bağlaç işleci uygulanır. Ardından her kuralın ardılındaki fonksiyonun sayısal değeri hesaplanır. Son olarak da ağırlıklı ortalama aracılığıyla kesin çıkış değeri hesaplanır [96].

Mamdani modelindeki durulaştırma aşaması, kurallara göre kırpma veya ölçekleme işleminden geçirilmiş ardıl bulanık kümelerden elde edilen bileşke bulanık çıktısını kesin bir değere dönüştürmektedir. Sugeno modelinde ise etkinleştirilen kurallar, bulanık girdi değişkenlerine (örneğin doğrusal fonksiyonlar yoluyla) bağlı kesin sonuçlar olduğundan durulaştırma gerekli değildir [88]. Sugeno modelinde kullanılan bulanık çıkarım mekanizması, girdi olarak giriş üyelik fonksiyonu bulanık kümesini kabul ederken çıkış olarak matematiksel fonksiyon değeri sunmaktadır.

Doğrusal çıktılara sahip bir Sugeno bulanık denetleyicisi, bulanık kural listesine göre uyarlanabilir doğrusal denetleyici olarak görülebilir [87]. Her bir kuralın ardıl kısmı,

öncülün doğrusal bir kombinasyonu olan bulanık çıkarımdır [88]. Bu çıkarımdan elde edilen kural ilgi dereceleri, ardılların doğrusal ara değerlemesinde ağırlık olarak kullanılır [87].

Çok girişli tek çıkışlı bir sistem göz önüne alındığında, $A_1^i, A_2^i, \ldots, A_M^i$ bulanık kümeleri girdi (öncül) ve $f^i(x_1, x_2, \ldots, x_M)$ fonksiyonu çıktı (ardıl) olmak üzere, Sugeno bulanık modelindeki $i$ numaralı bulanık kural $R^i$ aşağıdaki gibi ifade edilebilir [100].

$$R^i: \text{EĞER } \left(x_1 = A_1^i \text{ VE } x_2 = A_2^i \text{ VE } \ldots \text{ VE } x_M = A_M^i\right) \text{ İSE } \left(y^i = f^i(x_1, x_2, \ldots, x_M)\right)$$

Bu modelde, kuralın öncülü tarafından belirtilen bulanık bölgeden ardılının kesin değerini uygun şekilde tanımlayabildiği sürece $f^i$ herhangi bir fonksiyon olabilir. Genellikle bir polinom olan bu fonksiyon, kurala özgü olup her kuralda farklılık gösterebilir. Eğer tüm kurallar için $f^i$ birinci dereceden bir polinom ise, ortaya çıkan bulanık modele birinci dereceden Sugeno bulanık modeli adı verilir. Eğer her kuralda $f^i$ bir sabit ise, sıfırıncı dereceden Sugeno bulanık modeli elde edilir. Sıfırıncı dereceden Sugeno bulanık modeli, her kuralın ardılı bir bulanık tekton ile (ya da önceden durulaştırılmış bir ardıl ile) belirtilen Mamdani bulanık modelinin veya her kuralın ardılı bir sabiti merkez alan basamak fonksiyonu ile belirtilen Tsukamoto bulanık modelinin özel bir durumu olarak görülebilir [101].

Sugeno modelini kullanan bir bulanık mantık denetleyicisinin çıkışı $\hat{y}$, kesin çıktıları olduğundan tüm kuralların bileşkesi ağırlıklı ortalama yöntemiyle elde edilir. Bu sayede, Mamdani modelinde ihtiyaç duyulan zaman alıcı durulaştırma sürecinden kaçınılmış olur [101]. Bilgi tabanındaki bulanık kural listesinde $N$ adet kural varsa, çıkarım yapılan $i = 1, 2, \ldots, N$ kuralları için ardılın sayısal değeri $y^i$, ilgili kural öncülünün doğruluk gücü $w^i$ ile ağırlıklandırılarak sistemin nihai çıktısı Eşitlik 5.1'te gösterildiği gibi hesaplanır [102].

$$\hat{y} = \frac{\sum_{i=1}^{N} w^i y^i}{\sum_{i=1}^{N} y^i} \tag{5.1}$$

Her kuralın ağırlığı Eşitlik 5.2'de ifade edildiği gibi hesaplanır [90].

$$w^i = \prod_{j=1}^{M} A_j^i \left( x_j \right) \tag{5.2}$$

$\prod$ sembolü Eşitlik 5.3'de verilen bulanık en düşük olanı alma (min) işlecini belirtmektedir [90].

$$\prod_{j=1}^{M} A_j^i \left( x_j \right) = \min_{1 \le j \le M} \left\{ A_j^i \left( x_j \right) \right\} \tag{5.3}$$

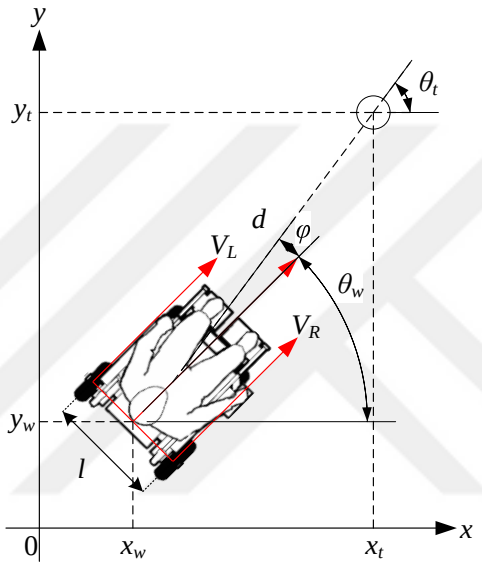
Burada $A_j^i \left( x_j \right)$, $x_j$ kesin girdisinin $A_j^i$ bulanık kümesindeki üyelik derecesidir [87].

Uygulamada, hesaplamayı daha da azaltmak için ağırlıklı ortalama yöntemi bazen ağırlıklı toplam yöntemi ile değiştirilebilir. Bununla birlikte bu basitleştirme, doğruluk gücü toplamı $\left( \sum_i w^i \right)$ birim değere yakın olmadığı sürece, üyelik fonksiyonu dilsel anlamlarının kaybına yol açabilmektedir [101].

Mamdani modeli ile Sugeno modelinde kullanılan kuralların öncül kısımları aynıdır. Kuralların ardıl kısımları ise Mamdami modelinde bulanık kümelerken Sugeno modelinde tektonlar (anlık değerler) veya bunların matematiksel fonksiyonlardır. Mamdami modelinin anlaşılması ve formüle edilmesi daha kolay olduğundan yaygın olarak kullanılmaktadır. Sugeno modeli ise hesaplama süreci verimli işlediğinden matematiksel analize dayalı sistemlerde kullanılması için daha uygundur [88]. Bu tez çalışmasında, hesaplama verimliliği ve analizde kolaylık önemli olduğundan Sugeno modeli tercih edilmiştir.

## 5.3 TEKERLEKLİ SANDALYENİN KİNEMATİK MODELİ

Sıradan bir tekerlekli sandalyenin, iki itici teker ve iki bağımsız sürüş tekerleği olmak üzere toplamda dört tekerleği bulunur. Birbirine paralel doğrultuda takılan itici tekerler, bir DA motor tarafından döndürülür. İtici tekerlerin hızlarına göre bir tekerlekli sandalye hareket ettirilerek yönlendirilebilir. Şekil 5.9, bir tekerlekli sandalyenin kinematik modelini göstermektedir [85].




Şekil 5.9. Tekerlekli sandalyenin kinematik modeli.

Şekil 5.9'da gösterilen $V_R$ ve $V_L$, tekerlekli sandalyenin sırasıyla sağ ve sol tekerleklerinin doğrusal hızıdır. Uzunluk değer $l$, iki itici tekerlek arasındaki mesafedir. Uzaklık değeri $d$, tekerlekli sandalyenin merkezi ile hedefi arasındaki Öklid mesafesidir. Açısal değer $\theta_w$, tekerlekli sandalyenin yönü ile $x$ ekseni arasındaki açıdır. Açı değeri $\theta_t$, tekerlekli sandalyenin tahrik merkezinden bitiş hedefine uzanan bağlantı çizgisinin $x$ ekseni ile arasındaki açıdır. Açı $\varphi$ ise $\theta_t$ ile $\theta_w$ açıları arasındaki farktır. Kartezyen sisteminin $x$ eksenindeki, $x_w$ ve $x_t$ sırasıyla tekerlekli sandalye tahrik sisteminin merkezi ve gidilmek istenen bitiş konumunun apsisidir. Benzer şekilde $y$ eksenindeki, $y_w$ ve $y_t$ sırasıyla tekerlekli sandalye tahrik sisteminin merkezi ve gidilmek istenen bitiş konumunun ordinatıdır.

Bu parametreler doğrultusunda, bir tekerlekli sandalyenin kinematik modeli açıklanabilir. Tekerlekli sandalyenin doğrusal hızı $V$ ve açısal hızı $\omega$, her bir itici tekerleğin hızına göre sırasıyla Eşitlik 5.4 ve Eşitlik 5.5 ile ifade edilebilir.

$$V = \frac{V_R + V_L}{2} \tag{5.4}$$

$$\omega = \frac{V_R - V_L}{l} \tag{5.5}$$

Tekerlekli sandalyenin istenen konuma hareketi boyunca yaşanan yer değişimlerinin aşamalı olarak yansıtılabilmesi için konum tahmini (dead-reckoning) süreci temel alınabilmektedir. Konum tahmini yönteminde, geçen süre boyunca önceki konum ve yönelim ile bilinen ya da tahmin edilen hızlar kullanılarak tekerlekli sandalyenin yeni konumu ve yönü elde edilir. Örnekleme zamanı $T$ için konum tahmini süreci, Eşitlik 5.6, Eşitlik 5.7 ve Eşitlik 5.8 ile verilen kinematik modelin ayrık denklemleriyle hesaplanabilir [85].

$$x_{w_T} = x_{w_0} + T\frac{V_{R_0} + V_{L_0}}{2}\cos\left(\theta_{w_0}\right) \tag{5.6}$$

$$y_{w_T} = y_{w_0} + T\frac{V_{R_0} + V_{L_0}}{2}\sin\left(\theta_{w_0}\right) \tag{5.7}$$

$$\theta_{w_T} = \theta_{w_0} + T\frac{V_{R_0} - V_{L_0}}{l} \tag{5.8}$$

Konum tahmini sürecine dayalı olarak tekerlekli sandalyenin temel hareketini bilmek mümkündür. Burada orijin noktası olarak belirlenen bir referans düğümüne göre tekerlekli sandalye, mevcut konum ve yönelim açısı $\left(x_w, y_w, \theta_w\right)$ ile temsil edilir. Şekil 5.9'da gösterildiği gibi tekerlekli sandalyenin geçerli konum ve yönelim açısından $\left(x_w, y_w, \theta_w\right)$ istenen konum ve yönelim açısına $\left(x_t, y_t, \theta_t\right)$ ulaşması için hesaplamalar yapılarak mesafe $d$ ve yönelim açısı $\varphi$ çıktıları elde edilir.

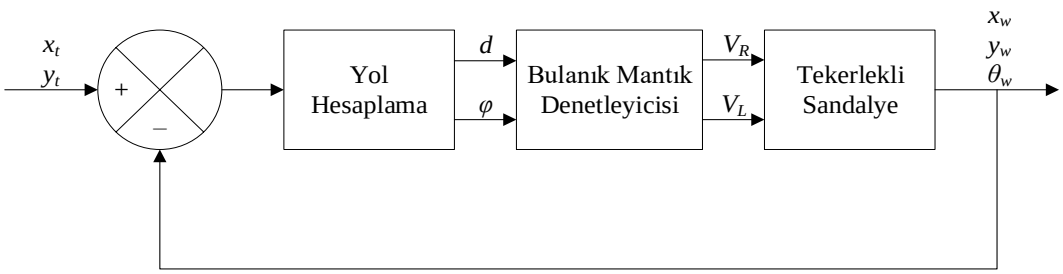
Bu çıktılar, Eşitlik 5.9, Eşitlik 5.10 ve Eşitlik 5.11 ile elde edilerek hareket denetiminde kullanılır [85].

$$d = \sqrt{\left(x_t - x_w\right)^2 - \left(y_t - y_w\right)^2} \tag{5.9}$$

$$\varphi = \theta_t - \theta_w \tag{5.10}$$

$$\theta_t = \tan^{-1}\left(\frac{y_t - y_w}{x_t - x_w}\right) \tag{5.11}$$

Seyir görevi için geliştirilen bulanık mantık denetleyicisinin mesafe $d$ ve açı yönelimi $\varphi$ olmak üzere iki girdisi, sağ tekerin hızı $V_R$ ve sol tekerin $V_L$ hızı olmak üzere de iki çıktısı vardır. Şekil 5.10, istenilen konuma ulaşmak için bulanık mantık denetleyicisi kullanan akıllı tekerlekli sandalye sisteminin kontrol döngüsüne ait blok diyagramı gösterilmektedir [103]. Bulanık mantık denetleyicisi, bulanıklaştırma, çıkarım ve durulaştırma temel bloklarından oluşur.

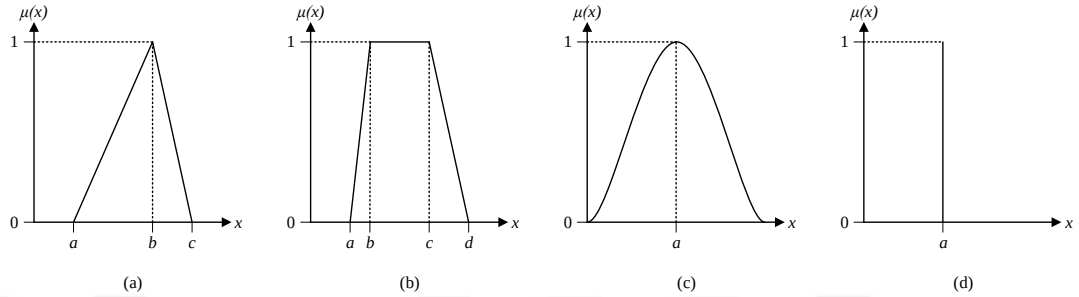



Şekil 5.10. Tekerlekli sandalyenin güzergâh takibinde kullanılan bulanık mantık kontrol sisteminin blok diyagramı.

## 5.4 ÜYELİK FONKSİYONLARI

Dilsel etiketler, bulanık kümeler kullanılarak temsil edilir ve kuralların öncülü ile ardılındaki özelliklerin tanımlanmasını sağlar [104]. Bir bulanık mantık denetleyicisinin girişi, bazı parametrelere karşılık gelen bir dizi kesin değerdir. Her bir giriş değeri için, üyelik fonksiyonları aracılığıyla uygun olan bulanık kümeye ait olma derecesi belirlenir [91]. Bu fonksiyonlar için doğrusal, parçalı doğrusal (örneğin

üçgensel ya da yamuksal), polinom, parçalı polinom şeridi (spline), üstel (örneğin Gauss normal dağılımı) veya kullanıcı tanımlı özel eğri (örneğin $\cos^2$) gibi mevcut problemin gerektirdiği herhangi bir uygun geometriye sahip olabilir [105]. Şekil 5.11'de en sık kullanılan bulanık küme biçimleri gösterilmiştir.




Şekil 5.11. En sık kullanılan bulanık küme biçimleri. a) Üçgensel b) Yamuksal c) Normal dağılım d) Tekton.

Bir bulanık küme, üyelik fonksiyonu ile karakterize edilir. Üyelik fonksiyonları simetrik olmak zorunda değildir [106]. Genel olarak, daha dar üyelik fonksiyonlarının kullanılması daha hızlı bir tepkiyle sonuçlanır, ancak daha büyük salınımlara, aşmaya ve yerleşme süresine neden olur [107]. Üçgensel üyelik fonksiyonu Eşitlik 5.12 ile, yamuksal üyelik fonksiyonu Eşitlik 5.13 ile, normal dağılım üyelik fonksiyonu Eşitlik 5.14 ile, tekton üyelik fonksiyonu ise Eşitlik 5.15 ile tanımlanabilir [90]. Tekton (singleton), taban uzunluğu sıfıra yaklaştıkça elde edilen üçgen şeklinin sınırlayıcı bir durumudur. Tektonlar, belirli bir ayrık noktada birim ve diğer her yerde sıfır olarak ifade edilir [88].

$$\mu(x) = \max\left(\min\left(\frac{x-a}{b-a}, \frac{c-x}{c-b}\right), 0\right) = \begin{cases} 0 & x < a \\ \dfrac{x-a}{b-a} & a \le x \le b \\ \dfrac{c-x}{c-b} & b < x \le c \\ 0 & x > c \end{cases} \tag{5.12}$$

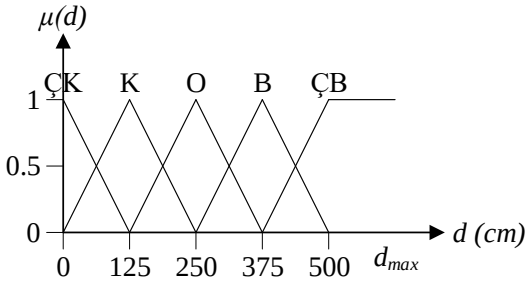
$$\mu(x)=\max\left(\min\left(\frac{x-a}{b-a},1,\frac{d-x}{d-c}\right),0\right)=\begin{cases}0 & x<a\\ \dfrac{x-a}{b-a} & a\le x<b\\ 1 & b\le x\le c\\ \dfrac{d-x}{d-c} & c<x\le d\\ 0 & x>d\end{cases} \tag{5.13}$$

$$\mu(x)=e^{-\frac{1}{2}\left(\frac{x-a}{\sigma}\right)^2} \quad \sigma>0 \tag{5.14}$$

$$\mu(x)=\begin{cases}1 & x=a\\ 0 & x\neq a\end{cases} \tag{5.15}$$

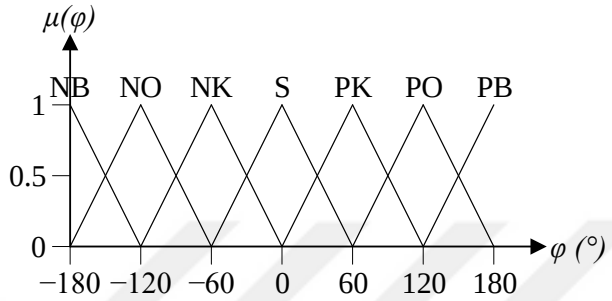
Bu çalışmada, giriş üyelik fonksiyonları için ikizkenar üçgen fonksiyonlar kullanılmıştır. Kullanılan sıfırıncı dereceden Sugeno modelinin bir getirisi olarak çıkış için üyelik fonksiyonlarına gerek duyulmamaktadır. Bu durum, tekton bulanık kümelerine sahip çıkış üyelik fonksiyonları kullanan Mamdani modeli ile eşdeğer olarak temsil edilebilmektedir.

Santimetre cinsinden $d$ mesafesi, 0 ile $d_{\max}$ arasında değişir ve $d_{\max}$ tekerlekli sandalye ile seyir alanında bulunan bir sonraki hedef arası en uzun mesafedir. Değişken mesafeye $(d)$ beş bulanık alt küme atanmıştır. Bunlar için, çok küçük (ÇK), küçük (K), orta (O), büyük (B) ve çok büyük (ÇB) notasyonları kullanılmıştır. Bu bulanık kümeler, Şekil 5.12'de gösterildiği gibi beş üçgen üyelik fonksiyonuyla [0 cm, 500 cm] söylem uzayını bölümlere ayırır.



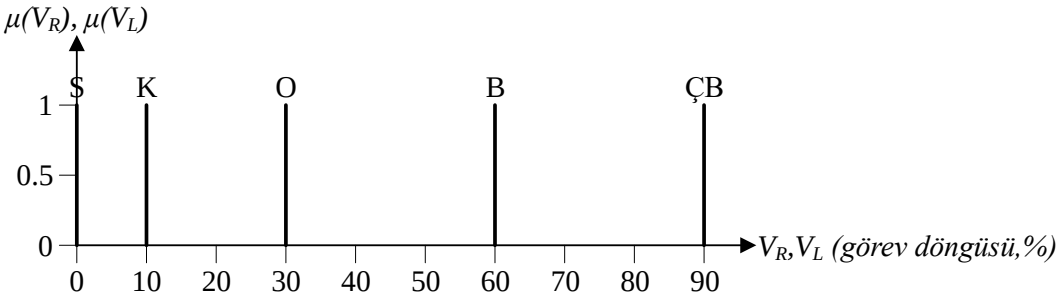

Şekil 5.12. Mesafe için giriş üyelik fonksiyonları.

Derece cinsinden $\varphi$ açısı, −180° ile +180° arasında sınırlandırılmıştır. Açı $(\varphi)$ ile yedi bulanık alt küme ilişkilendirilmiştir. Bunlar için, negatif büyük (NB), negatif orta (NO), negatif küçük (NK), sıfır (S), pozitif küçük (PK), pozitif orta (PO) ve pozitif büyük (PB) notasyonları kullanılmıştır. Bu bulanık kümeler, Şekil 5.13'te gösterildiği gibi yedi üçgen üyelik fonksiyonuyla [−180°, +180°] söylem uzayını bölümlere ayırır.




Şekil 5.13. Açı için giriş üyelik fonksiyonları.

Bu çalışmada, sıfır dereceli Sugeno modeli kullanılmıştır [97]. Şekil 5.14'te gösterildiği gibi Sugeno modelini kullanan bulanık mantık denetleyicisinin çıktıları, Mamdani modelinde kullanılan tekton fonksiyonları ile temsil edilebilir. Hız için belirlenen tektonlar, sıfır (S), küçük (K), orta (O), büyük (B) ve çok büyük (ÇB) notasyonları için sırasıyla %0, %10, %30, %60 ve %90 görev döngüsü (duty cycle) değerlerine konumlandırılmıştır [108]. Tekerleklerin $V_R$ ve $V_L$ hız değişimine karşılık gelen darbe genişlik ayarlı (DGA) sinyalin görev döngüsü oranları, simülasyon testleri yapılarak belirlenmiştir.




Şekil 5.14. Hız için Mamdani modelindeki eşdeğer üyelik fonksiyonları.

## 5.5 BULANIK KURAL LİSTESİ

Bulanık çıkarım mekanizması için üyelik fonksiyonlarının kullanımına ve şekline bağlı olarak bulanık kural listesinin tanımlanması gerekir. Detaylandırılan üyelik fonksiyonlarına göre, değişken girdi değerlerinin kombinasyonu çıktı değişkeninin bir değeri ile ilişkilendirilir.

Bulanık sistemde kullanılan kurallar, tekerlekli sandalye kullanıcısının yolda değişim belirdiğinde vereceğini tepkileri yakından takip edecek şekilde seçilmiştir. Bu kurallar, tekerlekli sandalyenin hedef konumu ile arasında kalan engelsiz mesafeye göre farklılık göstermektedir. Akıllı tekerlekli sandalye sisteminin sol ve sağ tekerlekleri için kullanıcının hem sürat hem de güvenlik talebini karşılayabilen 35 adet EĞER–İSE bulanık kuralı belirlenmiştir. Bulanık mantık denetleyicisinin tasarımında temel alınan kural listesi Çizelge 5.2'de verilmiştir [85].

Çizelge 5.2. Dilsel çıkarım tablosu.

| $d \setminus \varphi$ | | $\varphi$ | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | NB | | NO | | NK | | S | | PK | | PO | | PB | |
| $d$ | ÇK | B | K | O | K | K | S | S | S | S | K | K | O | K | B |
| | K | ÇB | O | B | K | O | K | K | K | K | O | K | B | O | ÇB |
| | O | ÇB | B | ÇB | O | B | K | O | O | K | B | O | ÇB | B | ÇB |
| | B | ÇB | B | ÇB | B | ÇB | O | B | B | O | ÇB | B | B | B | ÇB |
| | ÇB | ÇB | B | ÇB | B | ÇB | B | ÇB | ÇB | B | ÇB | B | ÇB | B | ÇB |

Mavi Renk : $V_L$ Yeşil Renk : $V_R$

Bulanık kuralların ilk ve sonuncusu aşağıdaki gibidir:

Kural 1: EĞER ($d$ Mesafesi Çok Küçük VE $\varphi$ Açısı Negatif Büyük) İSE ($V_L$ Hızı Büyüktür VE $V_R$ Hızı Küçüktür)

Bu kurala göre tekerlekli sandalye, sağ tarafında kalan bir sonraki hedefi ile arasında çok küçük mesafe olduğundan, sol motoru hızlı, sağ motoru ise yavaş sürülerek sağa doğru keskin dönüş yapar.

Kural 35: EĞER ($d$ Mesafesi Çok Büyük VE $\varphi$ Açısı Pozitif Büyük) İSE ($V_L$ Hızı Büyüktür VE $V_R$ Hızı Çok Büyüktür)

Bu kurala göre tekerlekli sandalye, sol tarafında kalan bir sonraki hedefi ile arasında çok büyük mesafe olduğundan, sağ motoru sol motorundan biraz daha hızlı sürülerek sola doğru döner.

Bulanık kuralların listelenmesinden sonra gelen adım, sıfırıncı dereceden Sugeno bulanık modeline uygun olarak bu kuralların bulanık çıkarım mekanizmasında sırayla değerlendirilmesidir. Her kuralın yerine getirilme derecesi (degree of fulfillment), bulanık "ve" bağlacının işleci kullanılarak hesaplanır. Etkinleştirilen kurallar, söz konusu kural için karşılık gelen yerine getirilme derecesi seviyesine göre birleştirilir [108].

Çelişkili bir durumda akıllı tekerlekli sandalye motorlarının ve motorları süren DA motor sürücüsünün vereceği tepki hızı çok önemlidir. Özellikle bulanık mantık denetleyicisinin temel aşamalarında tercih edilen yöntemler, yörüngenin seyri üzerinde büyük bir etki göstermektedir [109]. Bu tez çalışmasında, bulanık "ve" bağlacının işleci olarak en düşük olanı alma (min) yöntemi seçilmiştir. Ayrıca birden fazla kural çıktısının olduğu durumlarda bileşke çıkışın hesaplanması için Sugeno modelinin gerektirdiği şekilde ağırlıklı ortalama yöntemi kullanılmıştır.

# BÖLÜM 6

# SONUÇLAR VE ÖNERİLER

Bu tez çalışmasında, akıllı tekerlekli sandalye sistemi oluşturmak için sıradan bir akülü tekerlekli sandalyeye eklenmesi gereken materyaller ve bunların kullanılma yöntemlerinden bahsedilmiştir. Ardından otonom kontrol yazılımının test edilmesinde kullanılan simülasyon yazılımı CoppeliaSim ayrıntılı olarak anlatılmıştır. Bulunulan konum ile gidilmek istenen konum arasındaki en kısa yolu belirlemeyi sağlayan A* algoritmasının atlama noktası arama optimizasyonu incelenmiştir. Akıllı tekerlekli sandalye sistemini kullanıcı ile etkileşim kurmasını sağlayan görsel arayüz açıklanarak, yazılımda kullanılan kinematik model ve ilişkili bulanık mantık denetleyicisi teorik olarak izah edilmiştir.

Bu tez çalışmasıyla sıradan bir akülü tekerlekli sandalye tam otonom hâle getirilmekte ve bulunduğu ortamda kullanabileceği güzergâhın kolayca belirlenmesi sağlanmaktadır. Tadilat veya onarım gibi bir durum ile karşılaşıldığında, ortamdaki ani değişimlerin akıllı tekerlekli sandalye sistemine bildirilebilmesi için yazılımda kullanılan temsilî harita verileri kablosuz ağ aracılığı ile çevrimiçi güncellenebilmektedir. Bu sayede, akıllı tekerlekli sandalye sistemi kullanılabilir yollar üzerinden alternatif bir güzergâh belirleyerek ilerlemesine devam edebilmektedir.

Bundan sonra yapılacak çalışmalarda, sistem uyarılarının iletilmesi için kullanılan kablosuz mikrofonlu kulaklık üzerinden alınan kullanıcı ses sinyallerinin işlenmesiyle insan-makine arayüzü daha ileri seviyede geliştirilebilir. Sistemin hareket kabiliyetini kısıtlayan kapı gibi engellerin açılması için akıllı tekerlekli sandalye sistemine yardımcı olacak robotik kollar eklenebilir.

Güzergâh belirlemesinde kullanılan atlama noktası araması, en kısa yolu hesaplarken harita temsilinin tümünü inceleme eğilimindedir. Bu durum, ilk hareket öncesinde kullanıcının bir süre beklemesine neden olmaktadır. Çözüm olarak güzergâhta değişiklik meydana geldiğinde takip edilecek yolu aşamalı olarak onaran D*, Incremental A* ve D* Lite gibi A* algoritmasının farklı uzantıları kullanılabilir [110–113]. Bu algoritmalar, bilinmeyen veya dinamik ortamlarda mobil robot navigasyonu için yaygın olarak kullanılmaktadır.

Kontrol sisteminin iyileştirilebilmesi için akıllı tekerlekli sandalyenin dinamik kısıtlamalarının dâhil edildiği daha karmaşık modellemeler geliştirilebilir. Örneğin yüksek öteleme ve dönüş hızı ile yapılan keskin dönüşlerde tekerler kararsızlığa düşebilmektedir [114]. Bilgiye dayalı bulanık bir sistem tasarlandıktan sonra, daha iyi performans elde etmek için optimizasyon algoritmaları kullanılabilir [115]. Bu aşamada makine öğrenmesinden de yararlanılabilir.

## KAYNAKLAR


1. Tzafestas, S. G., "Research on Autonomous Robotic Wheelchairs in Europe", ***IEEE Robotics And Automation Magazine***, 8 (1): 4–6 (2001).
2. Bromley, I., "Tetraplegia and Paraplegia: A Guide for Physiotherapists, Sixth Edition", ***Elsevier***, Philadelphia (2006).
3. Al Sibai, M. H. ve Abdul Manap, S., "A Study on Smart Wheelchair Systems", ***International Journal Of Engineering Technology And Sciences (IJETS)***, 4 (1): 25–35 (2015).
4. Leaman, J. ve La, H. M., "A Comprehensive Review of Smart Wheelchairs: Past, Present, and Future", ***IEEE Transactions On Human-Machine Systems***, 47 (4): 486–499 (2017).
5. Simpson, R. C., "Smart Wheelchairs: A Literature Review", ***Journal Of Rehabilitation Research And Development (JRRD)***, 42 (4): 423–436 (2005).
6. Bourhis, G., Moumen, K., Pino, P., Rohmer, S. ve Pruski, A., "Assisted Navigation for a Powered Wheelchair", ***IEEE Systems Man And Cybernetics Conference (SMC)***, 553–558 (1993).
7. Connell, J. ve Viola, P., "Cooperative Control of a Semi-Autonomous Mobile Robot", ***IEEE International Conference On Robotics And Automation (ICRA)***, 1118–1121, Cincinnati (1990).
8. Leaman, J., La, H. M. ve Nguyen, L., "Development of a Smart Wheelchair for People with Disabilities", ***IEEE International Conference On Multisensor Fusion And Integration For Intelligent Systems (MFI)***, 279–284, Baden-Baden (2016).
9. Levine, S. P., Bell, D. A., Jaros, L. A., Simpson, R. C., Koren, Y. ve Borenstein, J., "The NavChair Assistive Wheelchair Navigation System", ***IEEE Transactions On Rehabilitation Engineering***, 7 (4): 443–451 (1999).
10. Simpson, R. C., Poirot, D. ve Baxter, F., "The Hephaestus Smart Wheelchair System", ***IEEE Transactions On Neural Systems And Rehabilitation Engineering***, 10 (2): 118–122 (2002).

11. García, J. C. G., Romera, M. M., Quintas, M. M. ve Ureña, J. U., "Positioning and Localization System for Autonomous Wheelchairs", ***IEEE Annual Conference Of The Industrial Electronics Society (IECON)***, 1555–1560, Sevilla (2002).
12. Martinez-Sala, A. S., Losilla, F., Sánchez-Aarnoutse, J. C. ve García-Haro, J., "Design, Implementation and Evaluation of an Indoor Navigation System for Visually Impaired People", ***Sensors***, 15 (12): 32168–32187 (2015).
13. Zhang, H.-M., Li, M.-L. ve Yang, L., "Safe Path Planning of Mobile Robot Based on Improved A* Algorithm in Complex Terrains", ***Algorithms***, 11 (4): 44 (2018).
14. Aversa, D., Sardina, S. ve Vassos, S., "Path Planning with Inventory-Driven Jump-Point-Search", ***AAAI Conference On Artificial Intelligence And Interactive Digital Entertainment (AIIDE)***, Santa Cruz (2015).
15. Zhou, K., Yu, L., Long, Z. ve Mo, S., "Local Path Planning of Driverless Car Navigation Based on Jump Point Search Method Under Urban Environment", ***Future İnternet***, 9 (3): 51 (2017).
16. Njah, M. ve Jallouli, M., "Wheelchair Obstacle Avoidance Based on Fuzzy Controller and Ultrasonic Sensors", ***International Conference On Computer Applications Technology (ICCAT)***, Sousse (2013).
17. Mac, T. T., Copot, C., Keyser, R. De, Tran, T. D. ve Vu, T., "MIMO Fuzzy Control for Autonomous Mobile Robot", ***Journal Of Automation And Control Engineering***, 3 (6): 65–70 (2015).
18. Kanoulas, E., Du, Y., Xia, T. ve Zhang, D., "Finding Fastest Paths on A Road Network with Speed Patterns", ***International Conference On Data Engineering (ICDE)***, Atlanta (2006).
19. Lamb, F., "Industrial Automation: Hands-On", ***McGraw-Hill***, USA (2013).
20. Helander, M. G., Landauer, T. K. ve Prabhu, P. V., "Handbook of Human-Computer Interaction, Second Edition", ***Elsevier***, Amsterdam (1997).
21. Jacko, J. A., "The Human-Computer Interaction Handbook: Fundamentals, Evolving Technologies, and Emerging Applications, Third Edition", ***CRC Press***, Boca Raton (2012).
22. Park, J. J. (Jong H., Chen, S.-C., Gil, J.-M. ve Yen, N. Y., "Multimedia and Ubiquitous Engineering", ***Springer***, Berlin (2014).

23. Flynn, R. R., "Computer Sciences, Volume 2: Software and Hardware", ***Macmillan Reference***, New York (2002).
24. Voland, G., "Engineering by Design, Second Edition", ***Pearson***, Harlow (2014).
25. Pulkki, V. ve Karjalainen, M., "Communication Acoustics: An Introduction to Speech, Audio and Psychoacoustics", ***John Wiley & Sons***, Chichester (2015).
26. Rayburn, R. A., "Eargle's the Microphone Book: From Mono to Stereo to Surround, A Guide to Microphone Design and Application, Third Edition", ***Focal Press***, Waltham (2012).
27. İnternet: "Depth Camera D435 - Intel® RealSense™ Depth and Tracking Cameras", **http://www.intelrealsense.com/depth-camera-d435** (2020).
28. McComb, G., "Robot Builder's Bonanza, Fourth Edition", ***McGraw-Hill***, USA (2011).
29. Ammari, H. M., "Mission-Oriented Sensor Networks and Systems: Art and Science, Volume 2: Advances", ***Springer***, Cham (2019).
30. İnternet: "UP Specifications - UPDatasheetV8.5", **http://up-board.org/wp-content/uploads/datasheets/UPDatasheetV8.5.pdf** (2020).
31. Monk, S., "Programming Arduino: Getting Started with Sketches, Second Edition", ***McGraw-Hill***, USA, 176 (2016).
32. Crisp, J., "Introduction to Microprocessors and Microcontrollers, Second Edition", ***Elsevier***, Oxford (2004).
33. Mandal, S. K., "Microprocessors and Microcontrollers", ***McGraw-Hill***, New Delhi (2014).
34. Langbridge, J. A., "Arduino Sketches: Tools and Techniques for Programming Wizardry", ***John Wiley & Sons***, Indianapolis (2015).
35. Hughes, J. M., "Arduino: A Technical Reference, A Handbook for Technicians, Engineers, and Makers", ***O'Reilly Media***, Sebastopol (2016).
36. Graf, R. F., "Modern Dictionary of Electronics, Seventh Edition", ***Newnes***, Woburn (1999).
37. Ben-Ari, M. ve Mondada, F., "Robotic Motion and Odometry", Elements of Robotics, ***Springer***, Cham, 63–93 (2018).
38. Yang Cheng, Maimone, M. ve Matthies, L., "Visual Odometry on the Mars Exploration Rovers", ***IEEE International Conference On Systems, Man And Cybernetics***, 1: 903–910, Waikoloa (2005).

39. Mohamed, S. A. S., Haghbayan, M.-H., Westerlund, T., Heikkonen, J., Tenhunen, H. ve Plosila, J., "A Survey on Odometry for Autonomous Navigation Systems", ***IEEE Access***, 7: 97466–97486 (2019).

40. Petschnigg, C., Breitenhuber, G., Breiling, B., Dieber, B. ve Brandstötter, M., "Online Simulation for Flexible Robotic Manufacturing", ***International Conference On Industrial Technology And Management (ICITM)***, 88–92, Oxford (2018).

41. Michel, O., "Webots$^{TM}$: Professional Mobile Robot Simulation", ***International Journal Of Advanced Robotic Systems***, 1 (1): 39–42 (2004).

42. Koenig, N. ve Howard, A., "Design and Use Paradigms for Gazebo, An Open-Source Multi-Robot Simulator", ***IEEE/RSJ International Conference On Intelligent Robots And Systems (IROS)***, 3: 2149–2154, Sendai (2004).

43. İnternet: "USARsim", **http://www.usarsim.sourceforge.net** (2020).

44. İnternet: "CoppeliaSim", **http://www.coppeliarobotics.com** (2020).

45. Alen, R. F. T. ve Silva, M. F., "Development and Simulation on V-REP of an Algorithm for the RoboCup@Work BNT", ***IEEE International Conference On Autonomous Robot Systems And Competitions (ICARSC)***, 315–320, Espinho (2014).

46. Bogaerts, B., Sels, S., Vanlanduit, S. ve Penne, R., "Connecting the CoppeliaSim Robotics Simulator to Virtual Reality", ***SoftwareX***, 11: 100426 (2020).

47. Rohmer, E., Singh, S. P. N. ve Freese, M., "V-REP: A Versatile and Scalable Robot Simulation Framework", ***IEEE International Conference On Intelligent Robots And Systems***, 1321–1326, Tokyo (2013).

48. Torres, F. M. L., "3D Pose Estimation for Bin-Picking Application", ***University of Lisbon***, Lisbon (2016).

49. Khalilov, J., "Interfacing Simulink-MATLAB with V-REP for Analysis and Control Synthesis of a Quadrotor", ***Middle East Technical University***, Ankara (2016).

50. Farias, G., Fabregas, E., Peralta, E., Torres, E. ve Dormido, S., "A Khepera IV Library for Robotic Control Education Using V-REP", ***IFAC-PapersOnLine***, 50 (1): 9150–9155 (2017).

51. Shamshiri, R. R., Hameed, I. A., Karkee, M. ve Weltzien, C., "Robotic Harvesting of Fruiting Vegetables: A Simulation Approach in V-REP, ROS and MATLAB", Automation in Agriculture - Securing Food Supplies for Future Generations, ***InTech*** (2018).
52. Quigley, M., Conley, K., Gerkey, B., Faust, J., Foote, T., Leibs, J., Wheeler, R. ve Ng, A., "ROS: an open-source Robot Operating System", ***IEEE International Conference On Robotics And Automation (ICRA)***, Kobe (2009).
53. İnternet: Coppelia Robotics AG, "CoppeliaSim User Manual Version 4.1.0", **http://www.coppeliarobotics.com/helpFiles/index.html** (2020).
54. Marc, F., Fumio, O. ve Nobuto, M., "Collision Detection, Distance Calculation and Proximity Sensor Simulation using Oriented Bounding Box Trees", ***International Conference On Advanced Mechatronics (ICAM)***, Asahikawa (2004).
55. Samet, H., "Neighbor Finding in Images Represented by Octrees", ***Computer Vision, Graphics, And Image Processing***, 46 (3): 367–386 (1989).
56. Jackins, C. L. ve Tanimoto, S. L., "Oct-Trees and Their Use in Representing Three-Dimensional Objects", ***Computer Graphics And Image Processing***, 14 (3): 249–270 (1980).
57. Chen, H. H. ve Huang, T. S., "A Survey of Construction and Manipulation of Octrees", ***Computer Vision, Graphics, And Image Processing***, 43 (3): 409–431 (1988).
58. Veenstra, J. ve Ahuja, N., "Line Drawings of Octree-Represented Objects", ***ACM Transactions On Graphics***, 7 (1): 61–75 (1988).
59. İnternet: "Bullet Physics Library", **http://www.bulletphysics.org** (2020).
60. İnternet: "Open Dynamics Engine", **http://www.ode.org** (2020).
61. İnternet: "Vortex Studio", **http://www.cm-labs.com/vortex-studio** (2020).
62. İnternet: "Newton Dynamics", **http://newtondynamics.com** (2020).
63. Kuffner, J. J. ve LaValle, S. M., "RRT-Connect: An Efficient Approach to Single-Query Path Planning", ***IEEE International Conference On Robotics And Automation (ICRA)***, 2: 995–1001, San Francisco (2000).
64. Hintjens, P., "ZeroMQ", Journal of Chemical Information and Modeling, ***O'Reilly Media***, Sebastopol (2013).

65. Reddy, K. K. ve Praveen, K., "Path Planning Using V-REP", ***International Journal Of Research In Engineering And Technology***, 02 (09): 94–97 (2013).
66. Jureczko, M., "Dynamic Analysis of the Exploration Robot Chassis in CoppeliaSim Software", ***Metody Komputerowe***, 49–52 (2020).
67. Ferguson, D. ve Stentz, A., "Field D*: An Interpolation-Based Path Planner and Replanner", Springer Tracts in Advanced Robotics, ***Springer***, Berlin, 239–253 (2007).
68. Beiglböck, K., "Programming GPS and OpenStreetMap Applications with Java", ***CRC Press***, Boca Raton, 125–126 (2012).
69. Sadhu, A. K. ve Konar, A., "Multi-Agent Coordination: A Reinforcement Learning Approach", ***John Wiley & Sons***, Hoboken, 10–15 (2021).
70. Dougherty, G., "Digital Image Processing for Medical Applications", ***Cambridge University Press***, 447 (2009).
71. Qidwai, U. ve Chen, C. H., "Digital Image Processing: An Algorithmic Approach with MATLAB", ***CRC Press***, 316 (2009).
72. Gonzalez, R. C. ve Woods, R. E., "Digital Image Processing", Third. Ed., ***Pearson*** (2008).
73. Hart, P. E., Nilsson, N. J. ve Raphael, B., "A Formal Basis for the Heuristic Determination of Minimum Cost Paths", ***IEEE Transactions On Systems Science And Cybernetics***, 4 (2): 100–107 (1968).
74. Zhang, Y., Li, L., Lin, H.-C., Ma, Z. ve Zhao, J., "Development of Path Planning Approach Based on Improved A-star Algorithm in AGV System", Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering, ***Springer***, Cham, 276–279 (2018).
75. Nilsson, N. J., "Principles of Artificial Intelligence", ***Morgan Kaufmann***, California (1980).
76. Koubaa, A., Bennaceur, H., Chaari, I., Trigui, S., Ammar, A., Sriti, M.-F., Cheikhrouhou, M. A. O. ve Javed, Y., "Background on Artificial Intelligence Algorithms for Global Path Planning", Robot Path Planning and Cooperation: Foundations, Algorithms and Experimentations, ***Springer***, Cham (2018).

77. Jingwen Li, Zhang, L., Zhao, Q., Wang, H., Lv, H. ve Feng, G., "A Heuristic Indoor Path Planning Method Based on Hierarchical Indoor Modelling", Communications in Computer and Information Science, Data Science, ***Springer***, Singapore, 467–476 (2018).
78. Dijkstra, E. W., "A Note on Two Problems in Connexion with Graphs", ***Numerische Mathematik***, 1 (1): 269–271 (1959).
79. Rabin, S. ve Silva, F., "JPS+ An Extreme A* Speed Optimization for Static Uniform Cost Grids", Game AI Pro 360: Guide to Architecture, ***CRC Press***, Boca Raton, 95–107 (2019).
80. Zeng, J., Qin, L., Hu, Y., Hu, C. ve Yin, Q., "Combining Subgoal Graphs with Reinforcement Learning to Build a Rational Pathfinder", Advanced Mobile Robotics Volume 2, ***MDPI***, Basel, 77–97 (2020).
81. Harabor, D. ve Grastien, A., "The JPS Pathfinding System", ***Annual Symposium On Combinatorial Search (SoCS)***, 207–208, California (2012).
82. Harabor, D. ve Grastien, A., "Online Graph Pruning for Pathfinding on Grid Maps", ***AAAI Conference On Artificial Intelligence***, 1114–1119, San Francisco (2011).
83. İnternet: La, W. G., "GitHub - Juhgiyo/EpPathFinding.Cs - A Jump Point Search Algorithm", **http://github.com/juhgiyo/EpPathFinding.cs** (2020).
84. İnternet: Jahn, M., "GitHub - Micjahn/ZXing.Net - Barcode Reader and Generator Library", **http://github.com/micjahn/ZXing.Net** (2020).
85. Baklouti, E., Amor, N. Ben ve Jallouli, M., "Autonomous Wheelchair Navigation with Real Time Obstacle Detection Using 3D Sensor", ***Automatika***, 57 (3): 761–773 (2016).
86. Slotine, J.-J. E. ve Li, W., "Applied Nonlinear Dynamics", Prentice-Hall, ***Prentice-Hall***, New Jersey (1991).
87. Nguyen, H. T. ve Sugeno, M., "Fuzzy Systems: Modeling and Control", ***Springer***, Boston (1998).
88. Reznik, L., "Fuzzy Controllers", ***Newnes***, Oxford (1997).
89. Zadeh, L. A., "Fuzzy Sets", ***Information And Control***, 8 (3): 338–353 (1965).
90. Zhang, H. ve Liu, D., "Fuzzy Modeling and Fuzzy Control", ***Birkhäuser***, Boston, 33–76 (2006).

91. Roy, S. ve Chakraborty, U., "Introduction to Soft Computing: Neuro-Fuzzy and Genetic Algorithms", ***Pearson***, Delhi (2013).
92. İnternet: Wierman, M. J., "An Introduction to the Mathematics of Uncertainty", **http://typo3.creighton.edu/fileadmin/user/CCAS/programs/fuzzy_math/docs/MOU.pdf** .
93. Mamdani, E. H., "Application of Fuzzy Algorithms for Control of Simple Dynamic Plant", ***Proceedings Of The Institution Of Electrical Engineers***, 121 (12): 1585–1588 (1974).
94. Fattouh, A. ve FadiFouz, "A Two-Stage Representation of Fuzzy Systems", ***International Journal Of Engineering Research And Applications***, 2 (3): 2660–2665 (2012).
95. Wang, L.-X., "A Course on Fuzzy Systems and Control", ***Prentice Hall***, Upper Saddle River (1997).
96. Espitia, H., Soriano, J., Machón, I. ve López, H., "Design Methodology for the Implementation of Fuzzy Inference Systems Based on Boolean Relations", ***Electronics***, 8 (11): 1243 (2019).
97. Takagi, T. ve Sugeno, M., "Fuzzy Identification of Systems and Its Applications to Modeling and Control", ***IEEE Transactions On Systems, Man, And Cybernetics***, SMC-15 (1): 116–132 (1985).
98. Sugeno, M. ve Kang, G. T., "Fuzzy Modelling and Control of Multilayer Incinerator", ***Fuzzy Sets And Systems***, 18 (3): 329–345 (1986).
99. Sugeno, M. ve Kang, G. T., "Structure Identification of Fuzzy Model", ***Fuzzy Sets And Systems***, 28 (1): 15–33 (1988).
100. Precup, R.-E. ve David, R.-C., "Introduction", Nature-Inspired Optimization Algorithms for Fuzzy Controlled Servo Systems, ***Elsevier***, 1–54 (2019).
101. Jang, J.-S. R., Sun, C.-T. ve Mizutani, E., "Neuro-Fuzzy and Soft Computing: A Computational Approach to Learning and Machine Intelligence", ***Prentice Hall***, Upper Saddle River (1997).
102. Chettibi, S. ve Chikhi, S., "FEA-OLSR: An Adaptive Energy Aware Routing Protocol for MANETs Using Zero-Order Sugeno Fuzzy System", ***International Journal Of Computer Science Issues (IJCSI)***, 10 (2): 136–141 (2013).

103. Omrane, H., Masmoudi, M. S. ve Masmoudi, M., "Fuzzy Logic Based Control for Autonomous Mobile Robot Navigation", ***Computational Intelligence And Neuroscience*** (9548482): (2016).
104. Cueva-Fernandez, G., Pascual Espada, J., García-Díaz, V. ve Gonzalez-Crespo, R., "Fuzzy Decision Method to Improve the Information Exchange in a Vehicle Sensor Tracking System", ***Applied Soft Computing***, 35: 708–716 (2015).
105. Shaw, I. S., "Fuzzy Control of Industrial Systems: Theory and Applications", ***Springer***, New York (1998).
106. Harris, J., "An Introduction to Fuzzy Logic Applications", ***Kluwer Academic Publishers***, Dordrecht (2000).
107. Siddique, N. ve Adeli, H., "Computational Intelligence: Synergies of Fuzzy Logic, Neural Networks and Evolutionary Computing", ***John Wiley & Sons***, Hoboken (2013).
108. Farooq, U., Hasan, K. M., Abbas, G. ve Asad, M. U., "Comparative Analysis of Zero Order Sugeno and Mamdani Fuzzy Logic Controllers for Obstacle Avoidance Behavior in Mobile Robot Navigation", ***International Conference And Workshop On Current Trends In Information Technology (CTIT)***, 113–119, Dubai (2011).
109. Špacapan, I., Kocijan, J. ve Bajd, T., "Simulation of Fuzzy-Logic-Based Intelligent Wheelchair Control System", ***Journal Of Intelligent And Robotic Systems: Theory And Applications***, 39 (2): 227–241 (2004).
110. Koenig, S. ve Likhachev, M., "D* Lite", ***AAAI Conference On Artificial Intelligence***, 476–483, Edmonton (2002).
111. Koenig, S. ve Likhachev, M., "Improved Fast Replanning for Robot Navigation in Unknown Terrain", ***IEEE International Conference On Robotics And Automation***, 968–975, Washington (2002).
112. Koenig, S. ve Likhachev, M., "Incremental A*", Advances in Neural Information Processing Systems, ***MIT Press***, 1539–1546 (2002).
113. Stentz, A., "The Focussed D* Algorithm for Real-Time Replanning", ***International Joint Conference On Artificial Intelligence (IJCAI)***, 2: 1652–1659, Montreal (1995).

114. Indiveri, G., Paulus, J. ve Plöger, P. G., "Motion Control of Swedish Wheeled Mobile Robots in the Presence of Actuator Saturation", ***RoboCup 2006: Robot Soccer World Cup X***, 35–46, Bremen (2007).

115. Fayaz, M., Ullah, I. ve Kim, D., "An Optimized Fuzzy Logic Control Model Based on a Strategy for the Learning of Membership Functions in an Indoor Environment", ***Electronics***, 8 (2): 132 (2019).

# ÖZGEÇMİŞ

Halil İbrahim ŞAHİN 1993 yılında Ankara'da doğdu; ilk ve orta öğrenimini aynı şehirde tamamladı. Alparslan Anadolu Lisesi'nden mezun oldu. 2012 yılında Karabük Üniversitesi Biyomedikal Mühendisliği Bölümü'nde öğrenime başlayıp 2017 yılında bölüm birinciliği ile mezun oldu.

## ADRES BİLGİLERİ


Adres : Karabük Üniversitesi
Balıklarkayası Mevkii / KARABÜK
Tel : (536) 597 82 28
E-posta : halilis@karabuk.edu.tr