
GAZİ ÜNİVERSİTESİ
Gazi
1926

DAİRESEL YAPIDA BİR OTOMATİK OTOPARK SİSTEMİNİN PROTOTİP TASARIMI VE UYGULAMASI

Ertuğrul ERDOĞDU

YÜKSEK LİSANS TEZİ
ELEKTRONİK - BİLGİSAYAR EĞİTİMİ ANABİLİM DALI

GAZİ ÜNİVERSİTESİ
BİLİŞİM ENSTİTÜSÜ

NİSAN 2105

Ertuğrul ERDOĞDU tarafından hazırlanan "DAİRESEL YAPIDA BİR OTOMATİK OTOPARK SİSTEMİNİN PROTOTİP TASARIMI VE UYGULAMASI" adlı tez çalışması aşağıdaki jüri tarafından OY BİRLİĞİ / ~~OY ÇOKLUĞU~~ ile Gazi Üniversitesi Bilişim Enstitüsü Elektronik Bilgisayar Eğitimi Anabilim Dalında YÜKSEK LİSANS TEZİ olarak kabul edilmiştir.

Danışman: Yrd. Doç. Dr. İsmail ATACAK

Bilgisayar Mühendisliği, Gazi Üniversitesi

Bu tezin, kapsam ve kalite olarak Yüksek Lisans Tezi olduğunu onaylıyorum/~~onaylamıyorum~~ .....

Başkan : Doç. Dr. İbrahim SEFA

Elektrik-Elektronik Mühendisliği, Gazi Üniversitesi

Bu tezin, kapsam ve kalite olarak Yüksek Lisans Tezi olduğunu onaylıyorum/~~onaylamıyorum~~ .....

Üye : Doç. Dr. Hamit ERDEM

Elektrik-Elektronik Mühendisliği, Başkent Üniversitesi

Bu tezin, kapsam ve kalite olarak Yüksek Lisans Tezi olduğunu onaylıyorum/~~onaylamıyorum~~ .....

Tez Savunma Tarihi: 30 / 04 / 2015

Jüri tarafından kabul edilen bu tezin Yüksek Lisans Tezi olması için gerekli şartları yerine getirdiğini onaylıyorum.

.....

Doç. Dr. Nurettin TOPALOĞLU

Bilişim Enstitüsü Müdürü

ETİK BEYAN

Gazi Üniversitesi Bilişim Enstitüsü Tez Yazım Kurallarına uygun olarak hazırladığım bu tez çalışmasında;

- Tez içinde sunduğum verileri, bilgileri ve dokümanları akademik ve etik kurallar çerçevesinde elde ettiğimi,
- Tüm bilgi, belge, değerlendirme ve sonuçları bilimsel etik ve ahlak kurallarına uygun olarak sunduğumu,
- Tez çalışmasında yararlandığım eserlerin tümüne uygun atıfta bulunarak kaynak gösterdiğimi,
- Kullanılan verilerde herhangi bir değişiklik yapmadığımı,
- Bu tezde sunduğum çalışmanın özgün olduğunu,

bildirir, aksi bir durumda aleyhime doğabilecek tüm hak kayıplarını kabullendiğimi beyan ederim.

.....

Ertuğrul ERDOĞDU

30/04/2015

DAİRESEL YAPIDA BİR OTOMATİK OTOPARK SİSTEMİNİN
PROTOTİP TASARIMI VE UYGULAMASI
(Yüksek Lisans Tezi)

Ertuğrul ERDOĞDU

GAZİ ÜNİVERSİTESİ
BİLİŞİM ENSTİTÜSÜ
Nisan 2015

ÖZET

Bu çalışmada üç eksende hareket edebilen bir robot sistemi kullanılarak, araçların katlı otoparktaki bölmelere otomatik olarak yerleştirilmesini sağlayan prototip bir uygulama gerçekleştirilmiştir. Uygulaması gerçekleştirilen otomatik otopark sistemi ile daha az alana daha fazla araç yerleştirilmesi amaçlanmaktadır. Park etmek için gelen aracın boş bölmelerden en uygun olanına park edilmesinde, en az enerji harcanması ve en kısa sürede park edilmesi temel kriterler olarak alınmışlardır. Taşıyıcı robot sisteminin eksensel hareketleri adım motorlar yardımıyla sağlanırken, bu adım motorların kontrolü PIC mikrodenetleyicisi tarafından gerçekleştirilmektedir. Bölmelerin boş ya da dolu olması durumu, kızılötesi yakınlık sensörleri ile mikrodenetleyiciye bildirilmektedir. Uygulamada bir web kamera ve bilgisayar yazılımı yardımıyla plaka tanımlama işlemi de gerçekleştirilmiştir. Yazılan bilgisayar programı tarafından, kamera ile fotoğrafı çekilen aracın plakası tanımlanmakta ve park süresine göre ücret hesaplaması yapılmaktadır. Plaka tanımlama işleminde Microsoft Office paketinde sunulan MODI .dll'inden yararlanılarak OCR (optik karakter tanıma) metodu kullanılmıştır. Bilgisayar ile PIC mikrodenetleyicisi RS-232 protokolü üzerinden haberleşmektedir.

Bilim Kodu	:	704.3.013
Anahtar Kelimeler	:	Otopark, adım motor, PIC, plaka tanıma, konum kontrolü
Sayfa Adedi	:	224
Danışman	:	Yrd. Dç. Dr. İsmail ATACAK

DESIGN AND IMPLEMENTATION OF A CIRCULAR TYPE AUTOMATIC PARKING SYSTEM

(M. Sc. Thesis)

Ertuğrul ERDOĞDU

GAZİ UNIVERSITY

INFORMATICS INSTITUTE

April 2015

ABSTRACT

In this study, a prototype robot system that moves on three axis was realized to automatically place vehicles to divisions in multi-storey car park. The implementation was aimed to park more vehicles to less space. Main criteria is to use minimum energy and minimum time to park a vehicle to most appropriate empty space available. While stepper motors supporting axis moves of carrier robot system, PIC microcontroller controls the step motors. Infrared proximity sensors reports to microcontrolles if a space is empty or occupied. License plate recognition was also realized with a web camera and computer software system. Camera takes the photo of vehicle, computer software recognize it and calculate the parking fee based on the time that vehicle parks in the car park. In license plate recognition, OCR (optical character reader) method was used with the help of the MODI.dll that Microsoft Office package presents. Computer and PIC microcontroller communicate with each other RS-232 protocol.

Science Code : 704.3.013
Key Words : Car park, stepper motor, PIC, plate recognition, position control
Page Number : 224
Supervisor : Asst. Prof. Dr. İsmail ATACAK

TEŞEKKÜR

Çalışmalarım boyunca değerli yardım ve katkılarıyla beni yönlendiren, kıymetli tecrübelerinden faydalandığım danışmanım Yrd. Doç. Dr. İsmail ATACAK'a, bilgisayar yazılımının hazırlanmasında her türlü desteği veren kardeşim Yusuf ERDOĞDU'ya, araştırmalarımda yardımlarını esirgemeyen meslektaşım Mustafa KOCATEPE'ye, mekanik aksam konusunda yoğun emeği geçen Muttalip KILIÇER'e, manevi desteklerini esirgemeyen sevgili kızım Zeynep ERDOĞDU'ya ve değerli eşime teşekkürü borç bilirim.

İÇİNDEKİLER

Sayfa

Sayfa

Sayfa

Sayfa

ÇİZELGELERİN LİSTESİ

Çizelge **Sayfa**

ŞEKİLLERİN LİSTESİ

Şekil **Sayfa**

Şekil **Sayfa**

Şekil **Sayfa**

Şekil **Sayfa**

RESİMLERİN LİSTESİ

Resim **Sayfa**

Resim **Sayfa**

SİMGELER VE KISALTMALAR

Bu çalışmada kullanılmış bazı simgeler ve kısaltmalar, açıklamaları ile birlikte aşağıda sunulmuştur.

Simgeler	Açıklama
%	Yüzde
°	Derece
°C	Santigrad
µF	Mikro farad
µSn	Mikro saniye
A	Amper
B	Mavi
D	Diyot
F	Kuvvet
G	Yeşil
gr	Gram
H_2O_2	Hidrojen peroksit
HCl	Hidroklorük asit
I	Akım
Jm	Eylemsizlik momenti
KB	Kilo bayt
Kg	Kilo gram
KHz	Kilo hertz
KΩ	Kilo ohm
L	Bobin
mA	Mili amper
Mb	Mega bit
mH	Mili Henry
MHz	Mega hertz
mm	Mili metre
ms	Mili saniye

Simgeler	Açıklama
N	Newton
nF	Nano farad
nm	Nano metre
Nm	Newton-metre
P	Basınç
P	Güç
pF	Piko farad
Q	Transistör
R	Direnç
R	Kırmızı
r	Yarıçap
RC	Direnç kondansatör
Sn	Saniye
T	Tork
t	Zaman
Tm	Moment
V	Volt
W	Güç
Θ	Teta
λ	Lamda
Ω	Ohm
ωn	Açısal hız

Kısaltmalar	Açıklama
A/D	Analog Dijital Çevirici (Analog Digital Convertor)
ABD	Amerika Birleşik Devletleri
AC	Alternatif Akım (Alternating Current)
ADC	Analog Dijital Çevirici (Analog Digital Convertor)
AFR	Fransız Sanayi Robotları Birliği (The Association Française de Robotique)
BDC	Fırçalı DC Motor (Brushed DC Motor)

Kısaltmalar	Açıklama
BLDC	Fırçasız DC Motor (Brushless DC Motor)
CCW	Saat İbresinin Tersi Yön (Counter Clock Wise)
CNC	Bilgisayarlı Sayısal Kontrol (Computer Numerical Control)
COM	Ortak (Common)
CP	Sürekli Güzergâh (Continuous Path)
CPU	Merkezi İşlem Birimi (Central Processing Unit)
CW	Saat İbresi Yönü (Clock Wise)
D/A	Dijital Analog Çevirici (Digital Analogue Convertor)
DC	Doğru Akım (Direct Current)
DIP	Çift Hatlı Paket (Dual In-Line Package)
EEPROM	Elektrik Enerjisi ile Silinebilen ve Programlanabilen Hafıza (Electrically Erasable Programable Read Only Memory)
EME	Elektro Manyetik Enerji (Electro Magnetic Energy)
GBS	Geri Besleme Sistemi
GND	Toprak, Şase (Ground)
HSM	Hibrid Adım Motor (Hybrid Stepper Motor)
I/O	Giriş/Çıkış (Input/Output)
IR	Kızılötesi (Infra Red)
IRQ	Kesme İsteği (Interrupt Request)
İTÜ	İstanbul Teknik Üniversitesi
JIRA	Japon Endüstriyel Robot Birliği (Japanese Industrial Robot Association)
LDR	Işığa Duyarlı Direnç (Light Depending Resistor)
LED	Işık Yayan Diyot (Light Emitting Diode)
LPT	Paralel Port (Line Printer Terminal)
MCLR	Reset Ucu (Master Clear)
N	Kuzey (North)
OCR	Optik Karakter Tanıma (Optical Character Recognition)
OSC	Osilatör (Oscillator)
PIC	Çevresel Arayüz Kontrolörü (Peripharel Interface Controller)

Kısaltmalar	Açıklama
PIC	Çevresel Arayüz Kontrolörü (Peripharel Interface Controller)
PLC	Programlanabilir Lojik Kontrolör (Programable Logic Controller)
PM	Sabit Mıknatıslı (Permanent Magnet)
POR	Enerji Uygulandığında Resetleme Özelliği (Power On Reset)
PSD	Pozisyon Duyarlılık Dedektörü (Position Sensitive Dedector)
PTP	Noktadan Noktaya (Point To Point)
PTS	Plaka Tanıma Sistemi
PVC	Poli Vinil Klorür (Polyvinyl Chloride)
PWM	Pals Genişlik Modülasyonu (Pulse Width Modulation)
R	Dönebilen (Rotary)
RAM	Rastgele Erişimli Bellek (Random Acces Memory)
RIA	Amerikan Robotik Enstitüsü (The Robotics Institue of America)
RISC	Azaltılmış Komut Kümesi Seti (Reduced Instruction Set Computing)
ROBOTEK	Robot Teknoloji Araştırma Ünitesi
ROM	Sadece Okunabilir Bellek (Read Only Memory)
S	Güney (South)
SCARA	Seçici Uyumlu Montaj Robot Kolu (Selective Compliance Automatic Robot Arm)
SCI	Seri İletişim Arayüzü (Serial Commnication Interfce)
TTL	Transistör Transistör Lojik (Transistor Transistor Logic)
TÜBİTAK	Türkiye Bilimsel ve Teknolojik Araştırma Kurumu
UART	Universal Asencronous Reciever Transmitter
USB	Uluslararası Seri İletişim Birimi (Universal Serial Bus)
UV	Morötesi (Ultra Violet)
VR	Değişken Relüktanslı (Variable Reluctance)

1. GİRİŞ

Bilgi ve teknoloji çağında yaşadığımız herkes tarafından kabul edilen bir gerçektir. Bu çağda insanoğlu hız ve zaman kavramlarına yetişebilmek için motorlu taşıtları bilhassa özel araçları daha fazla kullanır hale gelmiştir. Trafikteki araç sayısındaki artışa bağlı olarak otopark sorunu da günden güne büyümektedir. Özellikle metropollerde araçların hızlı, doğru ve güvenli bir şekilde park edilmesi hem trafiğin rahatlaması bakımından hem de mal ve can güvenliği bakımından oldukça önem kazanmıştır. Ayrıca teknolojik gelişmelere bağlı olarak yaşam kalitesinin artması, insanların daha fazla konfor ve kolaylık talep etmesine yol açmaktadır.

İnsanlar yaşam alanları arasında araçlı ya da araçsız olarak yer değiştirirler. Bu yer değiştirme işlemi gerçekleşirken araçlarından faydalanıyorlarsa elbette araçlarını park halinde bırakacakları otoparklara ihtiyaç duyarlar. Taşıtlar, gün içindeki zamanlarının ortalama 1,5-2 saatlik dilimini kent trafiğinde hareketli olarak harcarken, ömürlerinin ise yaklaşık %90'ını park halinde geçirirler [1].

Geleneksel (klasik) otoparklar hantal, çok fazla arazi işgal eden ve araç sahibi için kullanımı zor demode yapılar haline gelmişlerdir. Günümüzde bu sorunun çözümünde ileri teknolojiden faydalanılan, zamandan tasarruf sağlayan, çevreci, ekonomik, otomatik otopark sistemlerii veya mekanik katlı otopark olarak bilinen sistemler kullanılmaktadır. Dünyadaki kalabalık birçok şehirde, sıkışık kent merkezlerindeki otopark sorununu çözmek amacıyla bu tip sistemler tercih edilmektedir.

Bu çalışmanın amacı, zaman, mekan ve enerji tasarrufu sağlayan, sürücülere daha fazla konfor, kolaylık ve güvenlik sunan, teknolojik, çevreci otomatik bir otoparkın prototip uygulamasını gerçekleştirmektir. Uygulaması gerçekleştirilen otomatik otopark sistemi ile daha az alana daha fazla araç yerleştirilmesi amaçlanmaktadır. Park etmek için gelen aracın boş bölmelerden en uygun olanına park edilmesi için, en az enerji harcanarak ve en kısa sürede park etmesi temel kriterlerdir. Taşıyıcı robot sisteminin eksensel hareketleri step motorlar yardımıyla sağlanırken, bu step motorların kontrolü PIC mikrodenetleyicisi tarafından kontrol edilmektedir.

Araçların yerleştirileceği bölmelerin boş ya da dolu olması durumu, kızılötesi yakınlık sensörleri ile mikrodenetleyiciye bildirilmektedir. Uygulamada bir kamera ve bilgisayar yardımıyla plaka tanımlama işlemi de gerçekleştirilmiştir. Bilgisayar ile PIC mikrodenetleyicisi RS-232 protokolü ile haberleşmektedir.

Bu çalışma 6 bölümden oluşmaktadır. Çalışmanın 2. bölümünde otoparkın tarihi gelişimi anlatılmış, klasik ve otomatik otoparklardan bahsedilmiş ve literatür taraması yapılmıştır.

3. bölümde elektromekanik taşıyıcı robot sisteminin bileşenleri olan motor çeşitleri (DC, servo, adım motor) ve güç aktarım sistemleri (kayış-kasnak, vidalı mil, kremayer dişli sistemleri) incelenmiş; robot sistemleri, plaka tanımlamada kullanılan yöntemler ve OCR hakkında bilgi verilmiştir.

4. bölümde PIC 16F877A mikrodenetleyicisi hakkında bilgi verilmiş ve özellikleri incelenmiştir.

Yapılan çalışmanın aşamaları; otopark tasarımı, elektromekanik taşıyıcı robot sistemi, elektronik kontrol devresi, mikrodenetleyici yazılımı ve bilgisayar yazılımı 5. bölümde anlatılmıştır.

6. ve son bölümde ise ulaşılan sonuçlardan, projenin gerçekte uygulanması ile elde edilcek avantajlardan ve geliştirilmesi yönünde önerilerden bahsedilmiştir.

2. OTOPARK SİSTEMLERİ

Bu bölümde otopark sorununa değinilmiş; otoparkların tarihi gelişimi, park etme yöntemleri, klasik otoparklar ve otomatik otoparklar hakkında bilgi verilmiştir. Ayrıca yapılan çalışma ile ilgili literatür taraması yapılmıştır.

2.1. Otopark Sorunu

Otopark, motorlu araçların trafik bakımından uygun olan ve belli bir süre bırakıldıkları açık veya kapalı alanlardır. Otoparklar araçların belirli bir düzen içinde park etmeleri ve trafiği sıkıştırmamaları için yapılmışlardır. Özellikle büyük şehirlerde artan araç sayısı ile orantılı bir şekilde, sürücülerin park yeri ihtiyacını karşılayamamaları sonucu ortaya çıkan araç dolaşım problemi ve bunun doğurduğu olumsuz etkilerin tamamı “otopark problemi” olarak ortaya çıkmaktadır (Yardım ve Ağırlıklı 2005). Bazı büyükşehir merkezlerinde durum o kadar kötüdür ki, yollarda araç trafiği için ayrılmış yol şeritlerden bir ya da bir kaçı otopark olarak kullanılarak trafiğin akışı engellenmektedir. Park yeri ihtiyacının iyice tanımlanmadığı durumlarda, sürücüler zamanlarının önemli bir kısmını taşıtlarını bırakacak otopark yeri arayarak geçirirler. Sürücülerin kaybolan zamanlarının yanında oluşturdukları bu park yeri arama trafiği, kentin genel trafiği üzerinde önemli bir yük oluşturur [1].

Ulaşım planlarının imar planlarıyla birlikte yapılması, kentin nüfusundaki hızlı artış, araç sayılarındaki hızlı artış kent yaşamının en büyük sorunlarından olan ulaşım problemlerini beraberinde getirmektedir. Bunların sonucunda, hızla büyüyen kentlerde otopark sorunu ortaya çıkmaktadır.

Günümüzde insan yaşamı ile taşıtlar iç içe girmiş durumdadır. İnsanın aracından inip başka bir faaliyete geçmesi için, aracını park etmesi gerekmektedir. Bu anlamda yerleşim alanlarından çalışma alanlarına, dinlenme ve eğlence alanlarından spor alanlarına kadar birçok yaşam alanı otoparksız olamaz. Otoparkların insan yasamı içinde böylesine önemli ve etkin bir yer oluşturmuşken, onları planlamada göz ardı etmek akılcı değildir.

Özellikle büyükşehir merkezleri; iş, ticaret, finans ve kültür aktivitelerinin oldukça yoğun olduğu yerlerdir. Buralardaki çok katlı binaları otoparksız düşünmek neredeyse imkânsızdır.

2.2. Otoparkın Tarihi Gelişimi

İlk kapalı otoparklar ani pratik ihtiyaçlar dolayısıyla yapıldı. Hem şahsi araçların hem de filizlenmekte olan kiralık araba ve taksi sanayisinin belli bir ısıya ve elverişsiz hava şartlarından korunmaya ihtiyacı vardı. İlk otoparklar sadece koruma ve bakım amacı ile yapılmadı. Araçlara ve bu araçların kullanıcılarına farklı hizmetler sunmaktan da sorumluydu; soyunma kabini, tuvalet, uzun süreli park olanağı, perakende yerleri, araba yıkama ve bakımı, kuaför dükkânı vb [2].

Amerika Birleşik Devletleri nüfusunun 63 milyonun altında olduğu 1890 yılında, ülkede at ve katır sayısı 20 milyon civarındaydı. 1900 yılında en büyük 23 ABD şehrinde 1 454 000 ahır vardı ve 1907 yılına kadar, diğer şehirleri bir yana bırakırsak yalnızca Chicago'da 61 000'den fazla at arabası vardı [2].

At arabaları hem temel ulaşım aracı hem de kirliliğin ana sebebi konumunda idiler. Hayvan dışkıları ve hayvan leşleri çok ciddi bir çevre kirliği oluşturuyordu. Bu durum birçok salgın hastalığın da görülmesine neden oluyordu. İşte böyle bir zamanda otomobiller tarih sahnesindeki yerlerini aldılar. Atlar kirlilik kaynağı olarak, otomobiller ise şehirleri yaşamak için daha temiz ve sağlıklı yerler haline getirecek çevreyle barışık araçlar olarak görüldü. [2].

Otomobillerin tarih sahnesinde yerini alarak kullanılmaya başlanmasıyla Amerika ve Paris gibi bazı Avrupa kentlerinde otopark sıkıntısı baş göstermeye başladı. İnsanlar araçlarını güvenli ve olumsuz koşullardan koruyacak yapılara ihtiyaç duydular. Önceleri At ahırları otoparklara dönüştürülmeye başlansa da daha sonraları sadece otopark olarak hizmet veren yapılar inşa edilmeye başlandı [2].

İlk kişisel elektrikli araç Iowa, Des Moines'de W. Morrison tarafından geliştirildi. 1910'dan önce üretilen arabaların birçoğu elektrik gücüyle çalışıyordu ve bu nedenle elektrik firmaları bu teknolojinin gelişimine doğal olarak destek veriyordu.

Öyle ki ilk otoparkların birçoğu elektrikli arabalar için inşa edildi. 1899 yılına kadar, jetonlu mekanizmalar geliştirilmeye çalışıldı. Jeton karşılığı, elektrikli araç sürücüleri belirli bir watt /saat boyunca otoparktan yararlanabildi [2].

1894'de Philadelphia'dan Henry Morris ve Pedro Salom taksi olarak piyasa sürülen Electrobat adlı elektrikli bir arabanın ticari olarak üretimine başladılar. Üretici firmanın adı Electric Vehicle Company idi. Electric Vehicle Company (Elektrikli Araç Firması) aynı zamanda, ABD'deki sadece kapalı otopark olarak hizmet vermesi için inşa edilmiş ilk yapının sahibiydi [2].

1900'de Seely Manufacturing Company başkanı, D.N. Seely tarafından birçok otopark istasyonu inşa edildi. Seely, Pittsburgh ve Allegheny County, Pennysylvania'da elektrikli otomobil şarj istasyonları inşası için birçok proje yürüttü [2].

Boston'da 1913 yılında inşa edilmiş Edison Electric Garage, elektrikli araç otoparkının en erken örneklerinden bir başkasıdır. Edison Electric Company elektrikli araç otopark ve benzinli araç otoparkı da içinde olmak kaydıyla 7 hektarlık alana 7 bina inşa ettirdi. Bu otopark inşa edildiği vakitte ABD'deki en modern otopark olarak kabul edildi. Tuğla ve mermerden yapılmış otoparkın 3 dış cephesi ve çatısı camla kaplıydı [2].

Elektrikli araçlar buharlı araçlıdan daha uzun süre kullanıldı ancak Henry Ford'un her keseye uygun, ekonomik fiyatta araba üretimine başlaması nedeniyle, benzinli araçlar hem buharlı hem de elektrikli araçlara oranla daha uzun süredir kullanılmaktadır. Yapılan ilk otoparklar her üç araba türünü de barındırırken, zamanla yeni inşa edilen otoparklar sadece benzinli araçların ihtiyaçların karşılamaya yönelikti [2].

2.3. Park Etme Yöntemleri

Genellikle araçların hareketlilik için kullandığı yol kenarına ve yol dışına park etmek üzere iki yöntem vardır.

2.3.1. Yol kenarına park etme

Bu park yönteminde, araç hareketi için ayrılmış karayollarının kenarına süreli ya da süresiz olarak araçlar park edilir. Yol kenarına park etme 3 sınıfa ayrılabilir.

- Yola paralel park etme
- Yola dik (90^o) park etme
- Açılı park etme

Bu yöntem İstanbul, Ankara gibi büyükşehir merkezlerinde sıklıkla kullanılan bir yöntemdir. İstanbul'da büyükşehir belediyesine ait İSPARK şirketi tarafından, ücretli olarak, sürücülere bu park etme yöntemi ile hizmet sunulmaktadır.

2.3.2. Yol dışı park etme

Bu yöntemde ise araçlar karayollarının dışındaki yapılara ait otoparklara, ücretli olarak işletilen otoparklara, boş arazilere veya katlı otoparklara park edilirler.

2.4. Klasik Otoparklar

Mekanik cihazların, otomasyon sistemlerinin ve robotların kullanılmadığı otoparklardır. Bu otoparklar açık alanda düz bir zeminde, yeraltında, yer üstünde, yol kenarında ya da sadece otopark olarak hizmet veren çok katlı yapılar şeklinde olabilir [3].

Klasik otoparkların başlıca dezavantajları şunlardır:

Verimsiz alan kullanımından dolayı büyük boyutlu olmaları: Park bölmelerine ulaşmak için gerekli olan yollar, rampalar, insanın rahatça hareket etmesini sağlayacak tavan yüksekliği boyutları kullanılan arsa büyüklüğünü artıran başlıca etmenlerdir [3].

Şehir merkezlerinde uygulanabilirliğinin sıkıntılı olması: İstanbul, Ankara gibi metropollerin göbeğinde otopark yapacak kadar büyük araziler oldukça pahalı olmakla beraber uygun arazi bulmak da oldukça zordur.

Görsel kirliliğe neden olmaları: Açık otoparklar ya da katlı otoparklar şehir merkezlerinde görsel bir kirliliğe neden olmakta ve şehir merkezlerinin estetiğini bozmaktadırlar.

Güvenlik problemi: Klasik otoparklara herkes girebildiği için araçlar ve sürücüler için güvenlik problemi söz konusudur. Hırsızlık, gasp, yaralama, araçların zarar görmesi vb. olayların meydana gelmesi olasıdır. Hele hele değnekçi diye tabir edilen kişilerin kontrol ettiği ve mafya tarafından işletilen açık otoparklar çoğunlukla metruk araziler üzerinde bulunmakta ya da terk edilmiş yapıların bahçeleri otopark olarak kullanılmaktadır [3].

Konforsuz olmaları: Sürücüler otoparka girdiklerinde araçları için uygun park yeri bulmaya çalışırlar. Bu durum hem zaman ve yakıt kaybına neden olurken, hem de arkalarında araç kuyruğu oluşmasına neden olur. Ayrıca araçlarını almak için geldiklerinde de araçlarını bulmaları gerekir. İnsanlar otopark içinde araçsız hareket ederken zararlı gazlara ve toza maruz kalırlar [3,4].

2.5. Otomatik Park Sistemleri

Otomatik otopark sistemlerini Yardım ve Ağırlıklı (2005) şu şekilde açıklamışlardır; “Kullanıcıların otopark içerisine girmeyip araçlarını bir kabul odasında bıraktıkları ve araçların otomatik taşıyıcılarla park hücrelerine istiflendiği akıllı sistemlerdir.” Bu sistemlerle park yeri kapasitesi istenilen şartlar dâhilinde en üst düzeye çıkarılabilmektedir. Sistemin temel prensibi, sürücülerin müdahalesi olmaksızın, araçların sisteme giriş yaptıkları kabul odalarından hareketli platformlarla alınıp, robot asansörler vasıtasıyla, önceden belirlenmiş park yerlerine götürülerek park edilmesidir. Tamamen otomatik olarak çalışan sistemde güvenlik, hız, konfor ve teknoloji, çalışma prensiplerini belirleyen unsurlar olarak öne çıkmaktadır [1].

2.5.1. Kaldıraçlı basit sistemler

Bir araç gelip park ettikten sonra onu yukarı kaldırarak veya aşağı indirerek yeni boş bir park yeri açar (Şekil 2.1). Hidrolik ve asansörlü olabilen bu sistemler iki veya üç araçlık olabilmektedir. Bazıları kolaylıkla yer değiştirebildiğinden fuarlarda, işyerlerinde, belediyelerce yol kenarlarında kullanılabilmektedir [5].

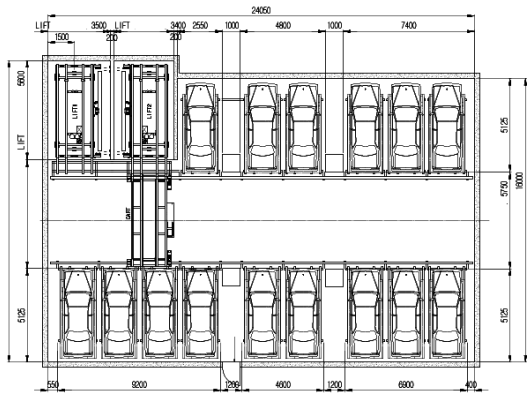
Kaldıraçlı basit sistemlerin yer değiştirebilir olması oldukça avantajlıdır, fakat az kapasiteli olmaları çok büyük bir dezavantajdır. Ticari olarak uygulanabilirliği tartışılır. Çok katlı yapılmaları mümkün değildir.

Şekil 2.1. Kaldıraçlı basit otopark sistemi

2.5.2. Yatayda ve dikeyde hareket eden sistemler

Bir koridorun ortasında bulunun raylı bir sistem vardır. Rayın üzerindeki palet, kabul bölgesinden aracı aldıktan sonra ray üzerinde hareket ederek her iki tarafında bulunan boşluklardan en uygun olanına aracı yerleştirir (Şekil 2.2). Bu sistem tek bir kat içindir. Katlı olarak uygulanmak istenirse her kat için bağımsız çalışan ünitelerin ayrı ayrı kurulması gerekir. Boylamasına ve enlemesine olarak dizayn edilebilir. Bu şekilde park alanları yatay düzlemdedir ve sadece yatay hareket vardır [5].

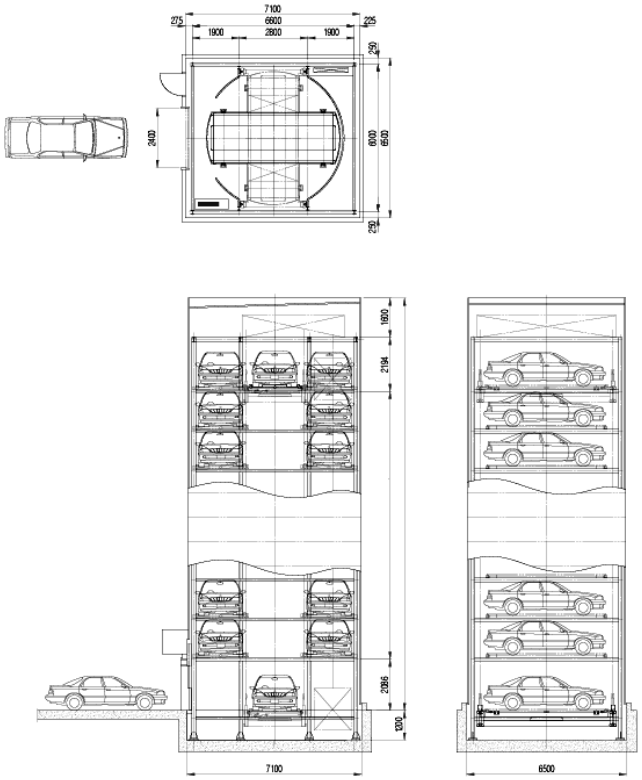
Sadece yatayda hareket eden sistemleri katlı olarak kurmak istediğimizde her kat için ayrı ve birbirinden bağımsız sistemlerin kurulmasının zorunluluğu oldukça maliyetlidir.



Şekil 2.2. Sadece yatayda hareket eden sistemler

Benzer bir sistem de kule tipi olarak kurulabilir. Bu tasarımda park alanları dikey iki blok şeklindedir ve bloklar arasında dikeyde aracı taşıyan bir asansör sistemi vardır (Şekil 2.3). Aracı kabul bölgesinden aldıktan sonra yukarı en uygun park alanına taşır ve yatay hareket ile sağda veya solda bulunan bir park alanına yerleştirir [5].

Bu sistemde en büyük dezavantaj taşıyıcı sistemin her kata sadece iki araç yerleştirebilir olmasıdır. İnşaat alanının küçük olması durumunda tercih edilebilirliği ise en önemli avantajıdır.



Şekil 2.3. Kule tipi otopark sistemi

2.5.3. Döner tip park sistemleri

Dönme dolap mantığına göre çalışır. En altta aracın park edebileceği boş bir tabla vardır. Araç park ettiğinde dönerek onu yakarı taşırken yerine boş bir tabla gelir. Birkaç araçlık olabileceği gibi kule tipinde yüksek olanları da vardır [6].

Benzer şekilde en büyük dezavantajı kapasite sorunudur. Birkaç araçlık olanı ticari olarak pek uygun değildir. Yüksek olanlarında ise (kat olarak düşünüldüğünde) her kata ancak iki araç yerleştirilebilir. Sistem hareket ettiğinde tüm araçlar yer değiştireceği için oldukça ciddi bir güce ihtiyaç vardır. Tabi ki büyük güç büyük enerji sarfiyatı anlamına gelmektedir.

2.5.4. Puzzel tip park sistemleri

Birkaç katlı olup dikey bloklar tek ya da toplu olarak yatayda sağa ve sola hareket edebilmektedirler. Bu hareketlerin sonucunda açılan boşluklar ile araç kabulü ya da teslimi yapılmaktadır. Boşluk açmak için dikey harekette yapılmaktadır (Şekil 2.4). Çalışması biraz karmaşıktır [7].

Yatay ve dikey hareketin sağlanabilmesi için yukarıdan aşağıya toplam bir blokluk boşluk olması zorunludur. Bu zorunluluğun kapasiteyi düşürmesi ciddi bir dezavantajdır. Bir aracın yerleştirilmesi veya teslim edilmesi çok basit olabileceği gibi hemen hemen her katın hareketini gerektirecek kadar karmaşık da olabilir. Bu da ciddi bir zaman kaybına ve enerji sarfiyatına neden olabilir.

Şekil 2.4. Puzzel tip otopark sistemi

Anlatılan sistemlerde kullanılan yöntemler kullanılarak daha başka birçok farklı otomatik park sistemi gerçekleştirilmiştir.

Bu çalışmada uygulaması gerçekleştirilen sistem yukarıda incelenen sistemlerde ki bazı dezavantajları gidermektedir. Burada hareketli sistem sadece taşıyıcı asansördür ve tasarımı da kolaydır. Hem dikeyde hem de dairesel olarak yatayda hareket edebilmektedir. Aracın kabulünde veya tesliminde sadece bir aracı taşıyacak kadar güce ve enerjiye ihtiyaç vardır. Çok katlı yapılabilir ve her kata çok sayıda araç yerleştirilebilir. Her kat için ayrı mekanizma kurmaya ihtiyaç yoktur.

2.6. Otomatik Otoparkların Avantajları

Arazi Tasarrufu: Yardım ve Ağırlıklı (2005), otomatik otoparkların diğer katlı otoparklara göre en önemli üstünlüğünü, çok küçük arsalarda bile maksimum kullanım alanı sağlayacak şekilde inşa edilebilmeleri olarak açıklamaktadırlar. Nitekim yıllar önce bu problemleri yaşayan Japonya, otomatik otoparklar üzerine ilk ve en büyük atılımların görüldüğü ülke olmuştur [1].

Klasik otoparkların çok katlı olması durumunda dolaşım için rampalı kat geçişleri kullanılır. Otopark içindeki sürüş koridorları, rampa geçişleri, yaya merdivenleri, yaya asansörleri, araçlar arası boşluklar ve sürücülerin manevra yapabilmesi için gerekli alanlar hesaba katıldığında bir araç için yaklaşık 30-35m^2 yapı alanı kat yükseklikleri ile birlikte hesaplandığında bir araç için yaklaşık 90m^3 yapı hacmi kullanılır. Otomatik otoparklarda ise araç dolaşımı için koridor ve rampalara ihtiyaç duyulmaksızın, katları sadece istenilen araç yüksekliklerine inşa ederek, alanlardan ve hacimden kazanç sağlanması, böylelikle otopark yapılamayacak kadar küçük olduğu düşünülen alanlarda dahi otopark yapılabilmesi, otopark için aynı şartlarda sağlanan yapı alanlarında, normal katlı otoparkların iki katından fazla kapasiteye ulaşabilmesidir (Bingöl ve diğerleri, 2010) [8].

İnşaat Maliyetinden Kazanç: 500 araçlık klasik otopark için kullanılan beton hacmi 4500m^3 iken, aynı kapasiteye sahip otomatik otoparkta bu hacim 2700m^3 olmaktadır. Buna kazı maliyeti, molozların taşınması, kullanılacak diğer inşaat malzemeleri ve işçilik dahil edildiğinde inşaat maliyeti oldukça azalmaktadır. Çizelge 2.1'de 500 araç kapasiteli her iki otopark için maliyet karşılaştırılması yapılmıştır [3].

Çizelge 2.1. İnşaat maliyeti açısından otomatik otoparklar ile klasik otoparkların karşılaştırılması

	Otomatik Otopark			**Klasik Otopark**		
Betonarme	Hacim (m^3)	Birim Maliyet ($/$m^3$)	Toplam Maliyet ($)	Hacim (m^3)	Birim Maliyet ($/$m^3$)	Toplam Maliyet ($)
Çimento harcı (Slurry walls)	1800	$200	$360 000	2600	$200	$520 000
Kazı (Excavation)	14000	$10	$140 000	22500	$10	$225 000
Döşeme (Slabs)	900	$500	$450 000	1900	$500	$950 000
Çeşitli (Miscellaneous)			$75 000			$60 000
Betonla yüzey bitirme (Concrete finishing)			$0			$120 000
Toplam betonarme maliyeti (Concrete total cost)			$1 025 000			$1 875 000

Güvenlik: Klasik otoparklar hırsızlık, saldırı, insanların veya araçların zarar görme ihtimalinin yüksek olduğu yerlerdir. Çünkü çoğunlukla sessiz ve yeterli aydınlatmadan yoksundurlar. Hele ki gece geç saatler olmuşsa bahsedilen olayların olma ihtimali daha da artar. Ayrıca manevra esnasında veya giriş çıkışlarda araçların zara görme olasılıkları vardır. Otomatik otoparklar kullanıcılar ve araçları için birçok konfor bileşeniyle birlikte, üst düzey bir güvenlik ortamı oluşturur. Çünkü sürücüler araçların park edildiği alanlara girmezler, herhangi bir insan da giremez. Ayrıca araçların park yerlerine yerleştirilmesi veya çıkarılması elektro-mekanik sistemler tarafından yapıldığı için kaza olma ihtimali çok büyük olasılıkla ortadan kaldırılmıştır [9].

Çevrecilik ve İnsan Sağlığı: Otomatik otoparklarda, park etme esnasında araç çalışmadığı için insan sağlığına zararlı gaz salınımı olmaz. Sürücüler içeri girmediği için, klasik otoparklarda kaçınılmaz olan zararlı gaz ve toz insan sağlığını etkilemez. Havalandırma ve aydınlatma için harcanan enerjiden tasarruf sağlanır. Ayrıca araçlar çalışmadığı için gürültü kirliliği de oluşmaz (Bingöl ve diğerleri 2010) [8].

Araçlar Arası Mesafe: Klasik otoparklarda, sürücülerin inip binebilmesi için ve manevra yapabilmesi için, araçlar arası mesafe en az 90cm olmalıdır. Otomatik otoparklarda bu mesafe 7cm'ye kadar inebilmekte ve böylece ciddi anlamda yer kazancı olmaktadır [3, 9].

Zamandan Tasarruf: Otomatik otoparklarda sürücü aracını kabul bölmesine bıraktıktan sonra manyetik kartını alır ve gider. Teslim almak için geldiğinde de bu kartı okutur, ödemesini yapar ve aracını teslim alır. Geleneksel otoparklarda ise sürücü uygun park yeri bulmak için, aracını park ettikten sona dışarı çıkmak için, aracını geri almak amacıyla aracının yanına gitmek için ve arcıyla tesis dışına çıkmak için bir hayli zaman harcar [9].

Personel Tasarrufu: Otomatik otoparklarda temizlik, güvenlik vb. işlerde çalıştırılan personel sayısı ya daha azdır ya da bu işlerde çalışacak personele hiç ihtiyaç duyulmaz. Böylece işletme giderleri azaltılır (Bingöl ve diğerleri 2010) [8].

Geri Dönüşüm ve Esneklik: Otomatik otoparkların inşaat hariç birçok parçası ve elektro-mekanik bileşenleri tekrar kullanılma ve yer değiştirme özelliğine sahiptir. Ayrıca bu otoparkların bileşenleri birçok yapıya uygulanabilecek esnekliğe sahiptirler [3].

Prestij: Otomatik otoparklar; insanlar, şehirler ve şirketler için bir statü sembolü veya gelişmişlik göstergesidir. Ayrıca bazı insanlar için gelişmiş teknolojiyi kullanarak kendilerini diğer insanlardan ayrıcalıklı görmektedirler [3].

2.7. Otomatik Otoparkların Dezavantajları

Birçok avantajının yanı sıra otomatik otoparklar aşağıda değinilen bazı dezavantajları vardır.

Kuyruk oluşması: Talebin yoğun olduğu saatlerde kuyruk oluşması söz konusudur. Klasik otoparklara aynı anda birden fazla aracın park etmesi mümkündür. Fakat otomatik otoparklarda bir taşıyıcı ancak bir araca hizmet verebilir. Kuyrukta bekleyen aracın park etme işleminin başlaması için önceki aracın işlemlerinin bitmesi gerekir. Bu olumsuzluk birden fazla taşıyıcının hizmet vermesi ile azaltılsa da yoğun saatlerde kuyruk oluşması ve sürücülerin beklemesi olası bir ihtimaldir [3].

Arıza Durumu: Taşıyıcılardan birinin arızalanması ciddi sıkıntılara neden olabilir. Daha kötüsü merkezi kontrol birimi (merkezi bilgisayar) arızalanırsa sistemin tamamı devre dışı kalabilir. Bu durumda sistemin kontrolünü devralacak yedek bir kontrol birimi bulundurulması neredeyse zorunludur [3].

Bakım: Sistemin mekanik ve elektronik bileşenlerine periyodik olarak bakım yapılmalı ve bazı bileşenler periyodik olarak değiştirilmelidir. Bu da işletme giderlerini artırmaktadır [3].

2.8. Literatür Taraması

Otomatik otoparkların geçmişi yaklaşık 30 yıl öncesine dayanmaktadır. Araştırmacılar teknolojinin ilerlemesi, şehir merkezlerinin yoğunlaşması, bireysel

araç kullanımının artması, özellikle şehir merkezlerinde araç trafiğinin artmasına bağlı olarak artan otopark sorununa modern ve teknolojik çözümler üretebilmek için otomatik otoparklar ile ilgili çalışmalar yapmışlardır. Bu sektörde proje üreten firmalar da değişik metotlar ile işlevini yerine getiren, otomatik otoparklar veya mekanik otoparklar geliştirmişlerdir.

Abboud N.W. 1994 yılında yaptığı “Automation of Parking Industry : A Strategic and Managerial Overwiev” isimli çalışmasında klasik ve otomatik otoparkların avantajlarını ve dezavantajlarını incelemiş, genel anlamda otomatik otoparklar ile klasik otoparkları karşılaştırmış, otomatik otoparkların maliyet, uygulanabilirlik, farklı yapılara adaptasyon, teknolojik olması gibi üstünlüklerini ortaya koymuştur. Ayrıca SARAH Co. Firmasının MATRA, IBM ve SCHNEIDER firmaları ile işbirliği yaparak ürettiği otoparkların mekanik yapısını, teknolojisini, güvenlik sistemlerini ve çalışma prensiplerini incelemiştir [3].

Lo L. 2008 yılında yaptığı “Automated Parking System” isimli çalışmasında PLC ile kontrol edilen, DC motor kullanılan bir otomatik park sisteminin prototip uygulamasını gerçekleştirmiştir. Bu çalışmada sürücü aracını bir paletin üzerine bıraktıktan sonra kontrol panelinde görüntülenen boş bir park yerini seçmekte ve sistem aracı otomatik olarak seçilen yere yerleştirmektedir [9].

Abu Hassan M. F. 2009 yılında yaptığı “Development of Automatic Parking System” isimli çalışmasında PIC ile kontrol edilen, Servo motor kullanılan 6 araçlık bir otomatik park sisteminin prototip uygulamasını gerçekleştirmiştir. Bu çalışmada dönen bir park sistemi ile sürücüye boş bir park alanı sağlanmakta ve sürücü aracını park edebilmektedir [4].

Bingöl ve diğerleri (2010) Isparta Süleyman Demirel Üniversitesinde yaptıkları “PLC Kontrollü Otomatik Katlı Otopark Sistemi” isimli çalışmalarında 9 araç kapasiteli, PLC ile kontrol edilen bir katlı otopark sisteminin prototip uygulamasını gerçekleştirmişlerdir. Bu çalışmada, her bir park yeri için kullanıcı şifresi atanmıştır ve bu şifre yardımı ile kullanıcılar araçlarını park yerlerine yerleştirip alabilmektedirler. Doğrusal bir ray üzerinde yatay hareket sağlanırken, taşıyıcı bir asansör ile dikey hareket sağlanmaktadır [8].

Sunitha ve diğerleri (2010) yaptıkları “Fuzzy Based Automatic Multi-Level Vehicle Parking Using Lab View” isimli çalışmalarında; AT89C52 mikrodenetleyicisi ve Lab View programı ile kontrol edilen, adım motor ve DC motorların kullanıldığı bir otomatik katlı otoparkın prototip uygulamasını gerçekleştirmişlerdir. Bu çalışmada Lab View programı içerisinde bulunan “Fuzzy System Designer” özelliği kullanılarak bulanık mantık denetleyicisinden faydalanılmaktadır. Araç büyüklüğü, aciliyet ve hizmet sınıfı gibi girdilere göre park yeri ataması yapılmakta ve ücret hesaplanmaktadır [10].

Mathijssen A. ve Pretorius A.J. (2006) yaptıkları “Verified Design of an Automated Parking Garage” isimli çalışmalarında otomatik park sistemlerinin daha güvenilir bir şekilde nasıl işleyebileceklerini incelemişlerdir. Otomatik park etme esnasında mekanik ve elektronik sistemlerin hatasız çalışmaları, araçların zarar görmemesi gibi güvenlik açısından önemli işlevleri kontrol edecek yazılım aşamalarını açıklamışlardır [11].

Reza ve diğerleri (2012) yaptıkları “Smart Parking System with Image Processing Facility” isimli çalışmalarında, tarak dişlerine benzer dilimli taşıyıcı bir palet (Resim 2.1.) yardımı ile aracı yine dilimli tabana sahip park bölmelerine yerleştiren otomatik bir otopark sisteminin hem prototipini geliştirmişler hem de Autodesk Inventor ve Comsol programları ile simulasyonunu yapmışlardır. Bu çalışmada elektromekanik sistemin kontrolü ve işleyişi bilgisayar tarafından yapılmaktadır. Bir webcam tarafından alınan görüntüler MATLAB programı tarafından işlenerek araçların plaka tanımlaması yapılmaktadır [12].

Resim 2.1. Dilimli taşıyıcılı palet sistemi

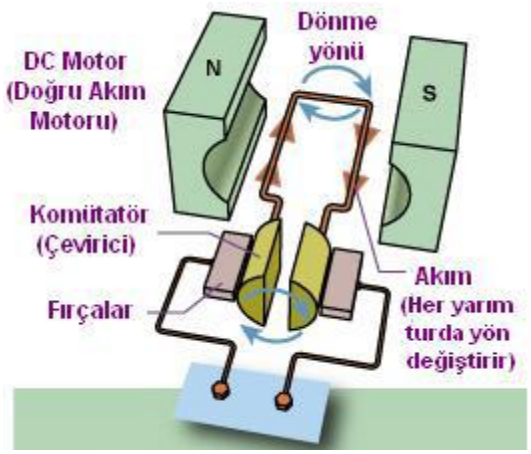
3. OTOPARK SİSTEMLERİNİ OLUŞTURAN BİLEŞENLER

Bu bölümde motorlar, güç ve hareket aktarımını sağlayan sistemler, robot sistemleri ve plaka tanıma aşamaları incelenmiştir. Yapılan çalışma için en uygun ve en verimli bileşenlerin seçimi için bu bilgilerden faydalanılmıştır.

3.1. DC Motor

Elektrik motorları; elektrik enerjisini hareket enerjisine çeviren makinelerdir. Genel anlamda çok çeşitli elektrik motorları bulunmaktadır. Elektrik motorları, çalışma prensiplerinine göre doğru akımla çalışan (DC) ve alternatif akımla çalışan (AC) olarak iki gruba ayrılabilir. Her elektrik motoru biri sabit (stator) ve diğeri kendi çevresinde dönen (rotor ya da endüvi) iki temel bileşenden oluşur.

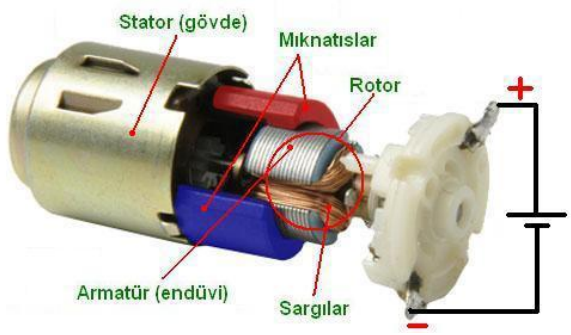
Elektriksel motor kavramı manyetik alan kuvvetinin elektrik akımıyla oluşturulması prensibine dayanmaktadır. Bir nüve üzerine sarılan çok sipirli bobin, tellerinin üzerinden akım geçirildiğinde bir manyetik alan oluşturur. N ve S kutuplarından oluşan bu manyetik alan, etki alanının içerisindeki manyetik cisimlere veya farklı manyetik alanlara tıpkı sabit mıknatısın gösterdiği etkiyi gösterir. Yani aynı kutuplu manyetik alanları iter; farklı kutuplu manyetik alanları çeker. Bu teoriden yola çıkan bilim adamları çekme itme kuvvetini ard arda koyarak ve bu hareketi dairesel harekete çevirerek ilk elektriksel motoru yapmışlardır [13].



Şekil 3.1. Elektrik motorunun çalışma prensibi

DC motorlar, statorda oluşturulan sabit manyetik alanın rotorda oluşturulan manyetik alanı itmesi ve çekmesi prensibine göre çalışır. Statorda kuzey-güney ekseninde oluşan sabit manyetik alana karşı, rotorda bu eksenden belli bir açıda kayık olarak yerleştirilen sargıda ikinci bir manyetik alan oluşturulur. Rotorun hareketi ile rotor sargısının stator sargısıyla aynı eksene gelmesi ve hareketin sona ermesini engellemek için rotor üzerinde birden fazla sargı oluşturulmuştur. Bu sargılar yine rotorun üzerindeki bir kolektörde toplanır. Kolektöre uygulanan gerilim, fırçalar yardımı ile aktarılır. Fırçalar, sabit eksende olduğu için rotor döndükçe gerilim uygulanan sargılar da değişecektir. Her defasında strator eksenine belli açıda manyetik alan oluşturan sargıya gerilim tatbik edildiğinden dönme sürekli devam eder [13].

DC motorların yol alma momentleri yüksektir ve devir sayıları geniş bir saha boyunca ayarlanabilir. Dönüş yönü değiştirilmek istendiğinde rotora uygulanan gerilimin polaritesi değiştirilir. Yani + ve – uçları ters bağlanır. Rotor (endüvi) akımı azaltılıp çoğaltıldığında motorun devri de değişecektir [13].



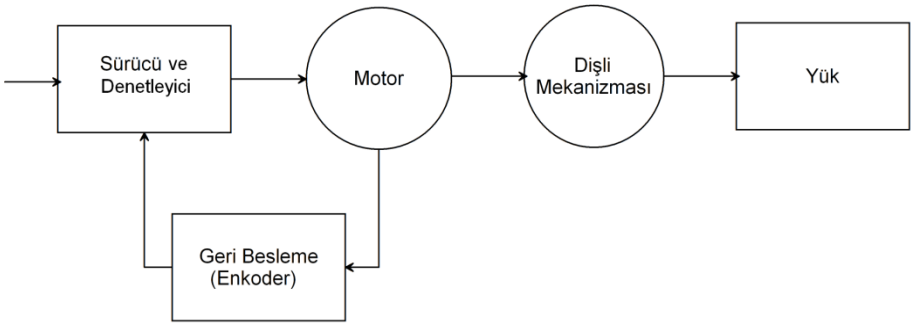
Şekil 3.2. Elektrik motorunun iç yapısı

DC motorlar daha çok sanayide kullanılırlar. Motorun dairesel hızı, uygulanan gerilim ile doğru orantılıdır. Çıkıştan alınacak moment ise bobin akım gücü ile doğru orantılıdır. Eğer motor hareketi hassas bir şekilde kontrol edilmek isteniyorsa dişli sistemi ve geri besleme kullanılabilir.

3.2. Servo Sistemler

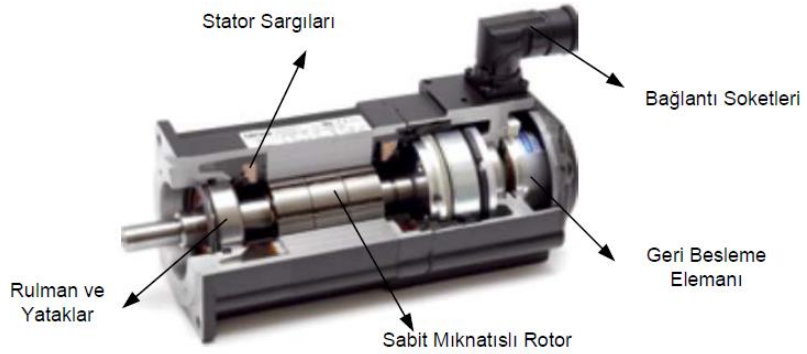
Servo sözcüğü Latincede “servus” sözcüğünden türetilmiş; “hizmetçi”, “köle”, “yardımcı” anlamına gelmektedir. Servo motorlar; konum kontrolü yapan, otomatik

kontrol sistemlerinde çok kullanılan özel motorlardır. Servo sistem Şekil 3,3'de görüldüğü gibi esas olarak bir kapalı çevrim kontrol sistemidir ve bu yapı içerisinde olmazsa olmaz eleman geri besleme elemanıdır. Bunun yanında diğer bir önemli eleman sürücü elemandır ve bu iki eleman olmadan servo sistem oluşturulamaz [14].



Şekil 3.3. Servo sistem prensip şeması

Servo motorlar endüstriyel uygulamalarda sistemin yapısına bağlı olarak bir ek elemanla birlikte kullanılırlar. Düşeydeki hareketlerde ya da yükten dolayı motorun dönmemesi için fren mekanizmasına, düşük hız - yüksek moment istenen durumlarda redüktöre, hız ve pozisyon kontrol uygulamaları için bir geri besleme elemanına, ağır şartlarda çalışıyor ve motor ısınıyorsa; soğutma sistemine ihtiyaç duyarlar [14].



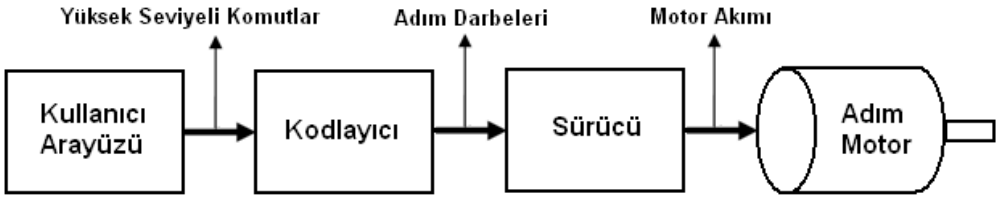
Şekil 3.4. Encoder geri beslemeli bir servo motor

3.3. Adım Motorlar

İlk olarak sayısal girişli adım motoru 1930 yılında denenmiştir. 1960'lara doğru elektronik teknolojisindeki gelişmeler, çok yüksek güçlü anahtarlama

transistörlerinin ve devrelerinin yapılabilme olanağını da beraberinde getirmiş ve bu gelişmeler sonrası adım motoru daha etkin bir şekilde kullanılmaya başlanmıştır. Günümüzde adım motorları endüstriyel alanlarda hassas konum kontrolü yapmak amacıyla kullanılmaktadır. Yazıcılar, çiziciler, disket sürücüler, hard disk sürücüler, kart okuyucular vb bilgisayar çevre cihazlarında bu elemanlardan yararlanılmaktadır. Ayrıca sayısal kontrol sistemlerinde, CNC (Computer Numerical Control) tezgahlarda, proses kontrol sistemlerinde, robot teknolojisinde ve uzay endüstrisine ait bir çok sistemde adım motorlar kullanılmaktadır [15].

Adım motoru, genel olarak elektrik enerjisini istenilen açıda dönme hareketine çevirebilen elektro-mekanik bir cihaz olarak tanımlanabilir. Adım motorlara Şekil 3.5'de gösterilen düzenek benzeri bir sürücü kombinasyonu ile elektrik enerjisi uygulandığında motor ve buna bağlı şaft, sabit açısal birimlerde (step - adım) dönmeye baslar. Darbe ile çalışan bu motorlara her darbe uygulandığında motor belli bir miktar dönmekte olup bu dönüş miktarına *adım* adı verilmektedir. Bazı motorların adımları 90 derece, daha hassas motorların ise 1.8° hatta 0.72° olabilmektedir. Uygun kontrolör sayesinde adım motorlarını yarım adım veya daha küçük ve mikro adım diye bilinen oldukça küçük adımlar şeklinde döndürmek mümkündür. Adım motorları otomatik kontrol devrelerinde genellikle geri besleme olmadan kullanılmaktadır. Bu motorların hassasiyetleri bir adımın %3 ve %5'i arasında değişmekte olup en önemlisi hatanın bir adımdan diğer adıma geçmemesidir. Böylece, geri besleme olmadan çok hassas motor pozisyon kontrol uygulamaları yapılabilmektedir [15].



Şekil 3.5. Adım motor kontrol blok şeması

DC servo motorlarda döner alan, rotor üzerindeki kollektör aracılığıyla dönüş hareketiyle birlikte rotor üzerindeki sarımların belirli bir sırayla devreye girmesiyle sağlanır. Senkron ve asenkron AC motorlarda ise döner alan stator ve rotorda

oluşan faz farkından yararlanılarak meydana getirilir. DC ve AC motorların aksine adım motorları, bobinlerine gerilim verildiğinde serbestçe dönmeye başlamazlar. Çünkü adım motoru, sargılarına uygulanan darbe gerilimi ile çalışır. Bir adım hareketi tamamlandıktan sonra sargı uçlarında enerji olmasına rağmen motor dönmez. Adım motoru tahrik devresinden bir adım darbesi aldığı zaman, motorun özelliğine göre rotor belli bir açıda döner ve bir sonraki adım darbesine kadar durur. Dolayısıyla motor milinin toplam açısal yer değiştirmesi, adım açısı ile verilen darbe sayısının çarpımına eşittir. Adım motorları; manyetik alanların karşılıklı etkileşimi (itme-çekme) prensibiyle çalışmaktadır. Sürücü durumundaki manyetik alan stratejik olarak yerleştirilmiş bobin gruplarının enerjilendirilip ardından enerjinin kesilmesi yoluyla döner. Bu dönen manyetik alan adım motorunun sabit mıknatıslı mil rotorunu da beraber çekerek döndürür ve hareket oluşur. Mikroişlemciler ya da lojik sistemler aracılığıyla bobinlerin sırayla devreye girmesi ile döner alan oluşturulur ve rotor, motoru kontrol eden mikroişlemci ya da lojik sistemin istediği hızda ve yönde dönmeye başlar [15, 16].

Sabit adım açısı adım motorlarını, hareket kontrolünde DC motorlara göre üstün kılan özelliktir. Adım motor karakteristiklerinde belirtilen tork ve hız değerleri aşılmadığı sürece bir bilgisayar sistemi, yolladığı adım komutlarını sayarak step motorun kaç adım attığını bilmekte ve böylece pozisyon hakkında kesin bilgi sahibi olmaktadır [15].

3.3.1. Adım motorların avantajları

Endüstride bir çok kullanım alanı bulunan adım motorların üstünlükleri şu şekilde sıralanabilir [15].

- Adım motorların açık çevrim davranışlarının ±1 adım doğruluk pozisyonuna sahip olmaları, yani kesin açısal mesafe tanımlanırsa motorun dönmesi uygun sayıda adımla (basamakla) kontrol edilebilir ve böylece mekanik sistemde milin hareketi yeterli ölçüde olur.
- Adım motorların yüksek torklarda düşük açısal hıza sahip olmalarından dolayı yüke yeterli momenti sağlayabilirler.

- Adım motorlar DC uyarmada geniş bir tutma torkuna sahiptirler. Yani adım motorların rotor hareketi sabitken otomatik kilitleme (self-locking) özelliği vardır. Bu durumda rotor sadece, terminal gerilimi zamanla değiştiği sürece hareket eder.
- Adım motorlarının çalışması faz sayısına göre sahip oldukları stator sargılarının ardı ardına uyarılması prensibine dayandığı için, sargı enerjilendirmeleri doğru yapıldığı sürece başlama, durma ve herhangi bir çalışma periyodunda yön değiştirme olaylarına hızlı tepki verebilirler.
- Adım motorları sayısal kontrollü sistemlere uygundurlar. Dolayısıyla mikrodenetleyici veya bilgisayar kontrollü uygulamalarda sorunsuzca kullanılmaları mümkündür.
- Hata yalnız adım hatasıdır (Genellikle adım basına %5 den daha azdır) ve bu nedenle birimsizdir. Adım motorların konum doğruluğu mükemmeldir.
- Motorda açık çevrim kontrolünün olması nedeniyle, takometre ve/veya encoderin motorda kullanılması gereksizdir. Geri besleme ile mil pozisyonunun tayin edilmesi elimine edilir. Buna bağlı olarak tasarımın maliyeti düşer.
- Motor bakımı kolay ve kullanım süresi uzundur. Bu nedenlerle maliyet düşmüş olur.
- Adım motorların ısınma gibi olumsuzluklardan gelen zararları azdır.
- Motor, bir güç kaynağı ve motor cevabını değerlendiren sürücü bir devre ile kontrol edilebilir.
- Adım motor sürücüleri ucuzdur ve kullanımları daha kolaydır.
- Fırçasız yapıya sahip olmaları nedeni ile ömürleri uzundur.
- Mevcut hata bir adımdan diğer adıma geçmez. Yani hatalar toplanarak büyümez.
- Mikro adım kontrolü kullanılarak dönüş hassasiyeti artırılabilir.

3.3.2. Adım motorların dezavantajları

Daha önce bahsedilen birçok avantajının yanında adım motorların bazı dezavantajları da vardır. Bunlar şöyle sıralanabilir.

- Adım açıları sabit olduğundan elde edilen hareket darbelidir. Bu nedenle bazı uygulamalarda bu motorların kullanımı uygun değildir.
- Sürtünmeden dolayı pozisyon kontrolünde hatalar oluşabilir.
- Aşırı yükler, açık döngülü kontrolde konum hatasına neden olabilirler.
- Çıkış güçleri sınırlıdır
- Momenti sınırlıdır. Isıile beraber verimleri düşer
- Geri beslemeli hız ve pozisyon kontrolü uygulamalarında kullanılmaları gerekirse ek donanımlara ihtiyaç duyulur.
- İyi kontrol edilmediğinde rezonans oluşabilir.
- Yüksek hızlarda çalıştırmak zor olabilir.
- Adım motorları genellikle diğer motortürlerinde daha pahalıdırlar.

3.3.3. Adım motorların diğer motorlarla karşılaştırılması

- Adım motorlar bir tam turu sürücü yardımıyla 50.000 adıma kadar bölebilirler. Servo motorların da bölme özelliği vardır. Fakat adım motor kadar kesin sonuç vermez. Ayrıca servo motor sürücüleri kendi içerisinde yuvarlama ve düzeltme yaptıkları için bu durum elde edilen sonucu etkiler.
- Servo motorlar üç gruba ayrılır; BDC servo (fırçalı DC servo), BLDC servo (fırçasız DC servo) ve AC servo. Bunlardan en kararlı olanı BDC servodur. Fakat fırçalı olması nedeniyle uzun ömürlü değillerdir. Adım motorların ömrü ise oldukça uzundur. BLDC servolar uzun ömürlü olmalarına rağmen BDC servo kadar hassas değillerdir. AC servo motorlar ise uzun ömürlü ve az problemli olmalarına rağmen hassas pozisyonlamada sorun olabilirler. Çünkü fazla yaylanma (salınım) yaparlar.
- Adım motorların sürücüleri hem ekonomik hem de uygulaması pratik, dijital kontrole uygun sürücülerdir.
- Adım motorlar pozisyonlamada mükemmel doğruluğa sahiptirler ve hataları toplayarak aktarmazlar.
- Yüksek tork gerektirmeyen işlerde açık çevrimde rahatlıkla kullanılabilirler. Servo sistemlerde ise geri besleme elemanı olazsa olmazlardandır.
- Başlama, durma ve yön değiştirme taleplerine çok hızlı tepki verirler.

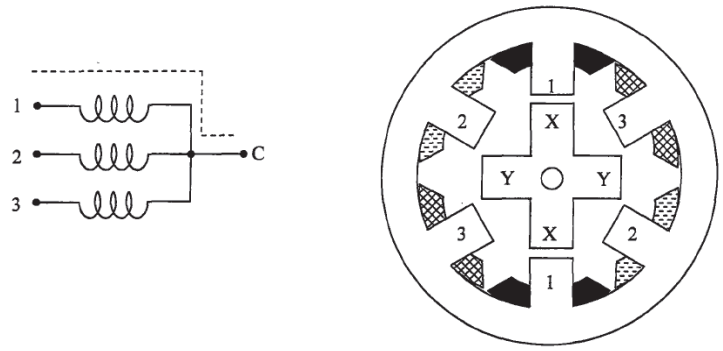
3.3.4. Yapısal olarak adım motor çeşitleri

Adım motorları yapılarına ve sargı bağlantı şekillerine göre sınıflandırılırlar. Motorun dâhil olduğu sınıf, kullanılacağı yerlere göre tercih edilme sebebi olmaktadır. Yapısal olarak adım motor çeşitleri şunlardır.

- Değişken relüktanslı adım motoru (VR)
- Sabit mıknatıslı adım motoru (PM)
- Hibrid adım motoru (HSM)

Değişken relüktanslı (VR) adım motoru

Bu tür motorlar sürekli mıknatıs içermezler. Yüksek motor momenti gerekmeyen ve endüstriyel olmayan alanlarda kullanım için uygundur. Değişken relüktanslı veya anahtarlamalı değişken relüktanslı adım motorlarının ortak bir uca bağlı üç ile beş arasında değişen sargıları bulunmaktadır. Şekil 3.6'da adım açısı 30^o olan değişken relüktanslı bir adım motorun statorundaki 3 faz sargının kesiti görülmektedir. Bu motorun rotorunda dört diş ve statorunda altı kutup bulunmaktadır. Aynı zamanda bu motorda her bir sargı karşılıklı duran kutuplar üzerine sarılmıştır. X ile işaretlenen rotor dişlileri, 1 numaralı faz sargısı enerjilendiği takdirde birinci sargıya doğru çekilirler. Rotor dişlilerinin birinci sargıya doğru çekilmelerinin nedeni, sargının ve rotorun etrafında meydana gelen manyetik akıdır. Böylece rotor, üzerinde meydana gelen bir moment sayesinde en kısa akı yolu oluşacak şekilde enerjilenmiş bulunan sargıların hizasına doğru hareket eder. 1 numaralı sargının enerjisi kesilir ve 2 numaralı sargı enerjilendirilirse motor saat ibresi yönünde hareket edecektir. Bu durumda Y ile işaretlenen rotor dişlileri 2 numaralı sargının hizasına geldiğinde motor saat ibresi yönünde 30^o'lik açısal yol kat etmiş olacaktır. Saat ibresi yönünde sürekli bir hareket elde etmek için, statorda bulunan sargıların enerjisi sıralı bir şekilde açılıp kapanması gerekir [16].

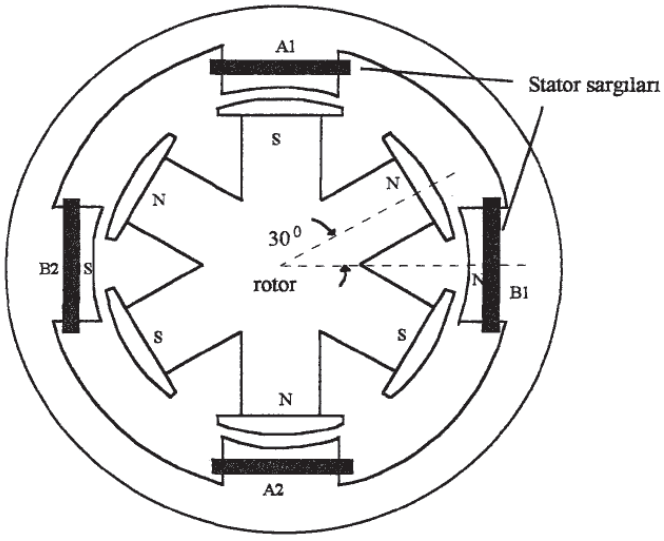


Şekil 3.6. Değişken relüktanslı adım motoru

Şekil 3.6'da görülen motor en temel değişken relüktanslı adım motorudur. Pratikte daha küçük adımlarla hareket edebilmek için, bu motorların daha çok stator kutupları ve rotor dişleri bulunmaktadır. Stator kutup sayısını artırmak için stator sargı sayısını 4 veya 5'e çıkarmak mümkündür ancak, genellikle buna çözüm olarak yani daha küçük adımlı hareket için dişli rotora göre çalışan dişli stator kutupları kullanılır. Bu yaklaşıma göre çalışan değişken relüktanslı adım motorlarının adım açısı yaklaşık olarak 1^o olabilmektedir [16].

Sabit mıknatıslı (PM) adım motoru

Değişken relüktanslı adım motoru ile benzerdir. Fakat sabit mıknatıslı adım motorunun rotoru N ve S olmak üzere sürekli mıknatıslı kutuplara sahiptir. Şekil 3.7'de sabit mıknatıslı bir adım motoru görülmektedir.



Şekil 3.7. 30°'lik adımlarla dönen sabit mıknatıslı bir adım motoru

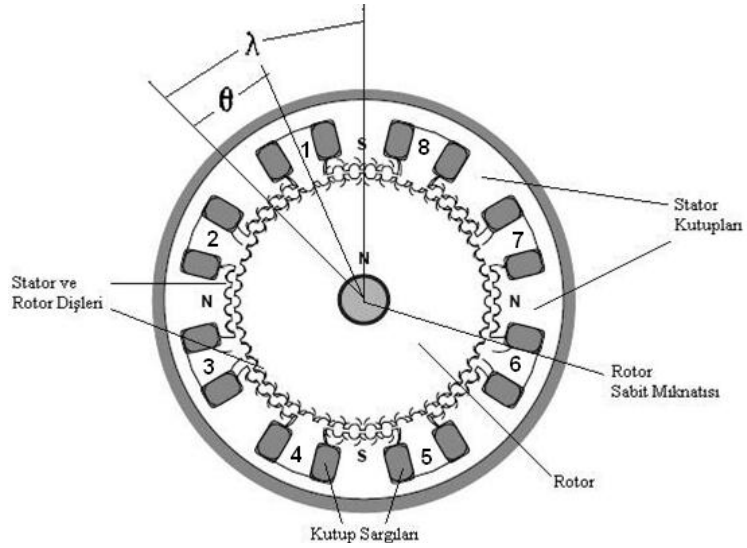
Sabit mıknatıslı adım motorlarında herhangi bir faz sargısı enerjilendiğinde rotor, stator sargısında oluşan manyetik kutuplanmaya uygun şekilde açısal bir dönüş hareketi yapar. Statorun diğer faz sargıları da belli bir sıraya göre enerjilendirilerek rotorun sürekli bir dönüş hareketi yapması sağlanır. Sargılara gerilim uygulandığında rotor adım hareketi yaptıktan sonra belli bir sabit pozisyonda kalır. Rotorun bulunduğu konumda üretilen momente “tutma momenti” denir.

Hibrid adım motorları

Hibrid adım motorları (HSM) ; sabit mıknatıslı (PM) ve değişken relüktanslı (VR) adım motorlarının birleştirilip geliştirilmesiyle ortaya çıkmış bir motor türüdür. HSM’ler kullanılacakları sistem ve yük özelliklerine bağlı olarak farklı yapılarda imal edilebilirler. Resim 3.1’de farklı hibrid motorlar görülmektedir [15].

Resim 3.1. Farklı hibrid adım motoru çeşitleri

Bu tip motorlar, aynı mil üzerine monte edilmiş birbirinin aynısı olan iki yumuşak demir endüviye sahiptir. Hibrid adım motorları yapısal olarak iki kısımdan oluşmaktadır. Şekil 3.8’de tipik bir hibrid adım motorunun kesitsel görünüşü verilmiştir [15].



Şekil 3.8. Hibrit adım motorlarında stator ve rotor yapıları

Şekil 3.8'de 8 (4-fazlı) stator kutbuna (40 stator dişli) ve 50 rotor dişine sahip bir hibrid adım motorunun kesit görünüşü verilmiştir. Burada ifade edilen *"θ"* yarım adım modunda stator kutupları arasındaki toplam adım açısını, *"λ"* ise tam adım modunda stator kutupları arasındaki toplam adım açısını vermektedir [15].

Hibrid adım motorlarında tipik olarak 8 stator kutbu bulunur. Her kutupta bulunan diş sayısı iki ila altı arasındadır. Ayrıca rotorun istenen konuma gelmesini sağlamak üzere mıknatıs akısının ilgili kutuplar üzerinden akısını desteklemek veya engellemek amacıyla stator kutuplarına sargılar da ilave edilmiştir [17].

Hibrid adım motorlarında iki farklı sargı kullanılır. Her bir sargı (faz), sekiz stator kutbundan dördünü dolaşır. A ve B sargıları 1, 3, 5, 7 kutupları üzerinde ise, C ve D sargıları 3, 4, 6, 8 kutuplarındadır. Her faza ait yakın kutuplar birbirleriyle zıt yönde sarılmışlardır. Öncekilerde olduğu gibi, hibrid adımlı bir motorun rotoru da, stator tahrik akımlarının uygun bir şekilde sıralanmasıyla adım adım hareket ettirilir [17].

Bu tip motorların rotoru sabit mıknatıslı olduğundan, her zaman için bir tutma veya kalıntı torku vardır. Motor, düşük hızla ivmelendiği takdirde saniyede 30 000 adıma kadar hızlarda çalışabilir. Tüm adım motorları içinde hibrid tip, her türlü doğrusal ve açısal konumlama sisteminde en yaygın kullanılan tiptir [17].

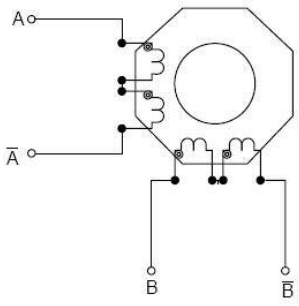
3.3.5. Adım motor sargı yapıları

Adım motorları endüstride genel olarak bağlantı şekli açısından çok kutuplu (unipolar) ve iki kutuplu (bipolar) olmak üzere 2 tipte üretilmektedirler.

Bipolar adım motorları

Bipolar adım motorlarında iki sargı bulunmaktadır ve bu sargıların uçları dört iletken halinde dışarıya çıkarılmıştır. Unipolar adım motorlarından farklı olarak, bipolar adım motorlarında orta uç yoktur. Orta uçların bulunmamasının avantajı sargı enerjilendiği zaman sargıdan geçen akım, sargının bir yarı bobininin sarımlarından değil bütün sarımlarından geçmesidir. Böyle olunca, aynı büyüklükte bir unipolar adım motoruna göre daha fazla moment üretir. Unipolar motorlara göre bipolar motorların dezavantajı daha karmaşık kontrol devrelerine ihtiyaç duymalarıdır [16].

Bipolar adım motorunun sargılarından çift yönlü akım geçmesi gerektiği için, sargılara uygulanan gerilimin yönü sürekli değişmelidir. H köprüsü olarak bilinen bir kontrol devresi sargı uçlarındaki kutuplanmayı değiştirmek için kullanılır. Bipolar adım motorlarında iki sargı bulunduğuna göre, bu motorları sürmek için iki köprü devresine ihtiyaç duyulur [16].

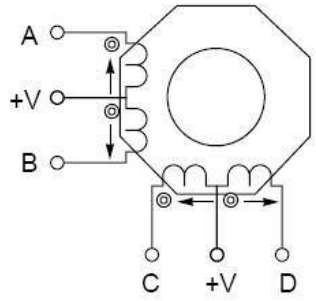


Şekil 3.9. Bipolar step motor sargıları

Ancak bipolar adım motorlarındaki akım yönünü değiştirme zorunluluğu ve bu nedenden dolayı kontrol devrelerinin nispeten karışık olması unipolar adım motorlarının ortaya çıkmasını sağlamıştır.

Unipolar adım motorları

Unipolar adım motorunun her bir stator kutbunda özdeş iki sargı bulunmaktadır. Bu tür sargılar akımın bir bobinden ters yönde sarılmış diğer bobine geçirilmesini sağlayarak motorun dönüş yönünün kontrol edilmesini kolaylaştırırlar.



Şekil 3.10. Unipolar adım motoru

Unipolar adım motorları 5 veya 6 uçlu olabilirler. 5 uçlu olanlarda iki sargının orta uçları birleştirilerek tek bir uç haline getirilmişlerdir. Unipolar adım motorlarının en büyük avantajı, sargılarda oluşan manyetik alanın yönü değiştirilmek istendiğinde sargılara uygulanan gerilimin yönünü değiştirmek yerine akım bir bobinden ters sarılmış diğer bobine kolaylıkla aktarılabilmesidir. Bu durum aynı stator kutbunda üst üste sarılmış özdeş iki sargı ile sağlanır. Tabi bu durumda aynı alana iki sargı sığdırılması gerekir ve bu durum sargıların iletken kesitinin azalmasına neden olur. Buna bağlı olarak da aynı boyuttaki bipolar adım motora göre daha az tork üretilir. Unipolar adım motorları, bipolar adım motorlarına kıyasla daha basit sürücü devreleri ile sürülebilmektedirler.

3.3.6. Adım motorların sürülmesi

Adım motorlarında rotorun dönüş hareketi için gerekli olan sargı akımları bir sürücü devresi ile sağlanır. Sürücüler kontrol biriminden aldıkları bilgilere göre adım motorunun istenilen yönde ve açıda dönmesi için motorun sargılarına akım darbeleri uygularlar. Sürücü devreleri birkaç transistör ve diyottan oluşacak kadar basit olabileceği gibi gelişmiş ve çok fonksiyonlu da olabilir. Gelişmiş sürücü üniteleri adım motorlarının kontrolünü kolaylaştıran kompakt devrelerdir.

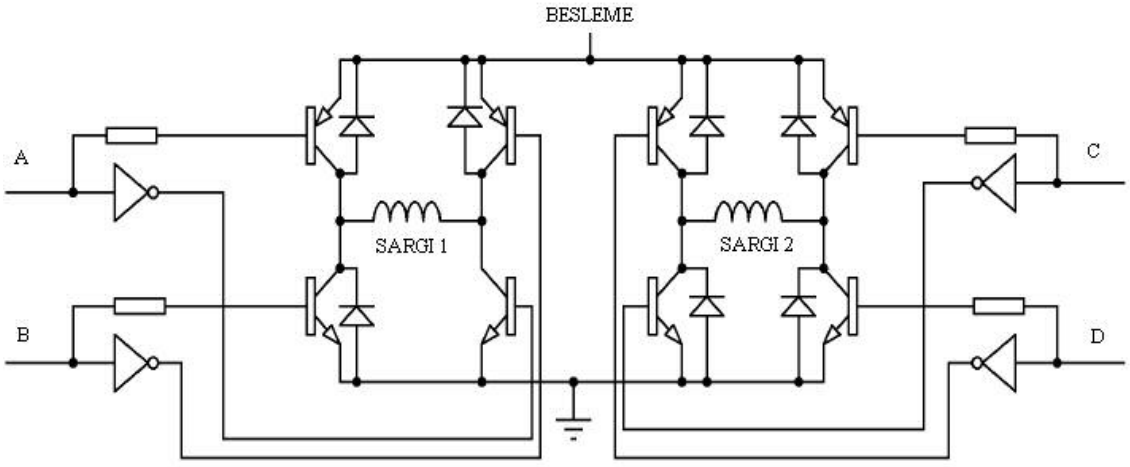
Bu tip sürücülerde mikrostep özelliği ile bir tam tur 200 ile 60 000 adıma bölünebilmekte, motora sağlanan akım ayarlanabilmekte ve motor dönüş yönü kontrol edilebilmektedir.

Adım motorları çeşitli yöntemlerle, çeşitli devrelerle, çeşitli sürücü entegreleri ile ve çeşitli sürücü üniteleri ile sürülebilmektedir. Burada bu yöntemlerden birkaçı incelenecektir.

Bipolar adım motoru sürücü devresi

Bipolar adım motorlarını sürmek için unipolar adım motorlarına göre daha karmaşık sürücü devrelerine ihtiyaç duyulur. Bunun sebebi bipolar adım motorunun sargılarından çift yönlü akım geçişinin olmasıdır. Bunun sağlanabilmesi için sargılara uygulanan gerilimin yönü de değişmelidir. H köprüsü olarak bilinen bir kontrol devresi sargı uçlarına uygulanan gerilimin polaritesini değiştirmek için kullanılır. Bipolar adım motorlarında iki sargı olduğu için, bu motorları sürmek için iki köprü devresi gereklidir.

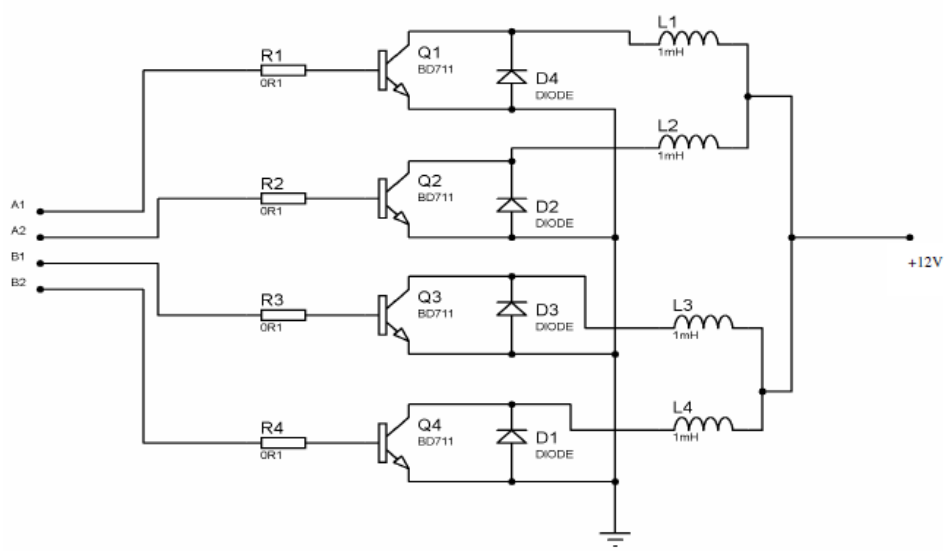
H-köprüsü temel sürme devresinde görüldüğü gibi tetikleme elemanı olarak transistörler kullanılmaktadır. İki sargı grubu için transistörlerle tasarlanmış köprü sürücü devresi Şekil 3.11'de gösterilmiştir. Bu devrede kontrol biriminden gönderilen lojik sinyaller A, B, C ve D ile gösterilen bipolar sargı uçlarını uyararak motor sargı (SARGI 1 ve SARGI 2) akım yönlerini değiştirmektedir [15].



Şekil 3.11. Bipolar adım motorun H-köprüsü ile sürülmesi

Unipolar adım motorun sürülmesi

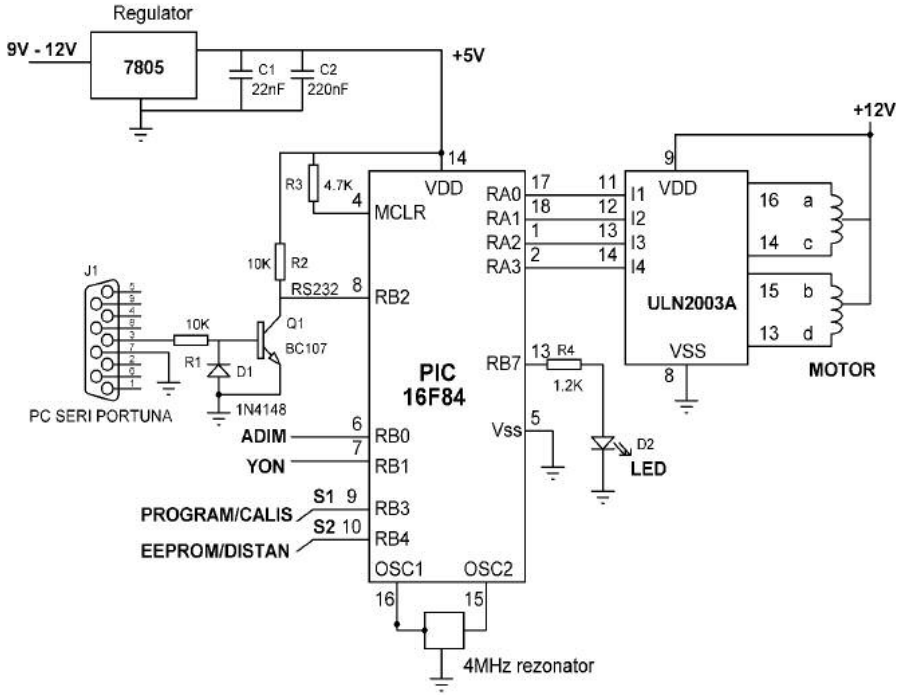
Daha önce bahsedildiği gibi unipolar adım motorlarını sürmek bipolar adım motorlarını sürmekten daha kolaydır. Bu tip motorlarda uygulanan uyarma metoduna göre her faz enerjilendirmesi için sadece bir uyarıcı elemanın kullanılması yeterli olmaktadır. Bu bilgiler ışığında 4 faz uçlu unipolar bir adım motoru için geliştirilmiş en basit sürücü devresi Şekil 3.12'deki gibidir [15].



Şekil 3.12. Unipolar adım motorunun transistörler ile sürülmesi

Adım motorların entegre devrelerle sürülmesi

Adım motorlar ULN2003, UCN5804 gibi entegreler ile sürülebilmektedir. Bu entegre devrelerin; tasarımı kolaylaştırmak, işlemleri basitleştirmek ve sürücü ebatlarını küçültmek gibi yararları vardır. Kullanıcıya sağladığı en büyük avantajlardan biri de bu entegrelerin dijital çıkışlı kontrol birimleri ile doğrudan kullanılabilmesidir. Adım motorlar 12V, 24V gibi DC gerilimleri ile sürülmekte fakat dijital sistemlerin çıkışlarında 5V gerilim bulunmaktadır. ULN2003 gibi entegre devreler yardımı ile giriş sinyali 5V olsa dahi, çıkışından sürülen motora 12V, 24V gibi sargı uyartım darbeleri uygulanabilmektedir. Şekil 3.13'de mikrodenetleyici ve ULN2003 entegresi ile bir adım motorun sürülmesine ilişkin devre şeması görülmektedir.



Şekil 3.13. Mikrodenetleyici ve ULN2003 entegresi ile bir adım motorun sürülmesi

Gelişmiş sürücü üniteleri

Günümüzde sistem performansını iyileştirici, parametre değerleri ayarlanabilir, gelişmiş özelliklere sahip, bilgisayar veya PLC gibi kontrol üniteleri ile haberleşen ve kompakt yapıya sahip birçok sürücü ünitesi ve sürücü kartı piyasada mevcuttur. Bu üniteler kullanıcın işini son derece kolaylaştırmakla birlikte güvenilirliği de artırmaktadır. Resim 3.2'de Robosan firması tarafından tedarik edilip satışa sunulan bazı sürücü üniteleri görülmektedir.

Resim 3.2. Bazı gelişmiş adım motor sürücü üniteleri

Robosan firması ürünlerinden ZM-2H606 adım motoru sürücüsü iki faz, 4,6 ve 8 telli step motorlar için üretilmiştir. Yüksek frekanslı giriş sinyallerini kabul edebilecek şekilde donatılmıştır. Yüksek akım kararlılığı, çok güçlü parazit önleme kabiliyeti, çok başarılı yüksek frekans performansı, yüksek başlangıç frekansı, giriş ve çıkış devresi izolasyonu, ayarlanabilir akım, kararlı çalışma, yüksek doğruluk ve düşük gürültülü çalışma gibi üstün özelliklere sahiptir. Ürün, gövdeye monte soğutuculu olarak sunulmaktadır [18].

3.3.7. Adım motorların parametreleri

Sargı Direnci: Adım motorunun bir fazı için stator sargısının DC direnç değeri olup *R* ile gösterilir ve birimi Ω'dur [15].

Sargı İndüktansı: Adım motorunun bir fazı için stator sargısının maksimum indüktans değeridir. *L* ile gösterilir ve birimi *mH*'dir [15].

Rotor Eylemsizliği: Rotor eksenlerine bağlı olarak değişen eylemsizlik momentinin değeridırı *JM* ile gösterilir ve birimi kg/m^2'dir [15].

Temel Adım Açısı: Uygulamada motorun bir fazı uyarıldığında meydana gelen açısal dönmeye denir. θa ile gösterilir ve derece (º) birimindedir. Değeri kullanılan adım motoru tipine göre değişir [15].

Akım: Manyetik devre doyumu, sıcaklık artışı vb. gibi değerler göz önünde bulundurulması durumunda tanımlanan nominal sargı akımıdır. IR ile gösterilir ve birimi A(amper)'dir [15].

Gerilim: Sargılardan nominal bir IR akımını geçişini sağlayan uygulanan gerilimin değeri olup *VR* ile gösterilir ve birimi *V(volt)*'tur [15].

Histerizis Hatası: Motorun saat yönü (CW- counter wise) ile saat yönünün tersinde (CCW- counter clock wise) dönmesi arasındaki tüm statik açısal hatalarındaki maksimum fark olup $\Delta\theta h$ ile gösterilir ve birimi derece (º)'dir.

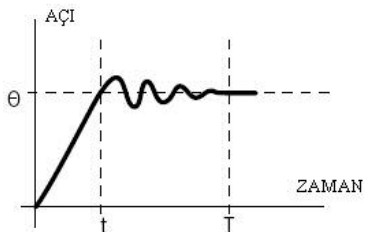
Tek Adım Tepkisi (Single Step Response): Motor fazlarından biri uyarılmış durumdaysa, motor kararlı bir adım konumundadır. Bu fazın uyartımı kesilip yeni bir faz uyartılırsa rotor bir adım atacaktır [17, 19].

Rotor konumunun zamana göre değişimi “tek adım tepkisi” olarak tanımlanır. Bir başka deyişle motorun girişine bir komut palsı (sinyali) uygulandıktan sonra motorun adımlara cevap vermek için gerekli olarak aldığı zamana (T) “tek adım tepkisi”, “adım tepkisi” ya da “cevap süresi” denir. Bu süre hem motor parametrelerine hem de motorun sürücü devresine bağlıdır ve yaklaşık olarak milisaniye civarlarındadır [17].

Tek adım tepkisi, motorun adım hareketinin hızını, tepkinin aşım ve salınım miktarını, adım açısının hassaslığını veren önemli bir parametredir. Adım motorlarından maksimum performans elde edebilmek için tek adım tepkisindeki aşım ve salınımların azaltılması ve yerleşme zamanının kısaltılması gerekmektedir. Bu nedenle tek adım tepkisinin iyileştirilmesi adım motorlarının kontrolünde çok büyük öneme sahiptir [17].

Motora uygulanan giriş darbeleri ile çıkış hareketi arasındaki senkronizasyonu bozmamak için, sürme esnasında iki darbe arasındaki süre, cevap süresinden daha kısa olmamalıdır. Dolayısıyla adım motorunun cevap süresinin kısaltılabilmesi, motorun daha hızlı adım atabilmesini sağlayacaktır [17].

Adım motorlarının cevap davranışlarında dikkat edilmesi gereken diğer bir nokta da yaptıkları aşımın (overshoot) ve osilasyonun miktarıdır. Adım motorları, bilgisayar sistemlerinde veya bilgisayarla kontrolü gerektiren hassas sistemlerde kullanıldığında aşım ve osilasyon, sistemi kötü yönde etkileyen ve önemli boyutta hatalara sebep olan bir davranıştır [17].



Şekil 3.14. Adım motoru için zamana bağlı tek adım tepkisi ve sönüm karakteristiği

Adım oranı : Bir saniyede rotorun yapabildiği adım sayısına adım oranı denir. Bu adım sayıları, tipik olarak saniyede 300 ila 800 arasındadır [15].

Doğruluk : Rotorun yaptığı her bir adımdaki hata miktarını adım açısı doğruluk parametresi gösterir. Bu parametre genellikle yüzde olarak verilir. Bir step motorun adım konumu, tasarım ve üretim sırasında bir araya getirilen birçok parçanın boyutları ile belirlenir. Bu parçaların boyutlarındaki toleranslar ve dahili sürtünmeler adımların nominal denge konumlarında da toleranslara neden olurlar. Bu durum belli bir konumdaki maksimum açısal hatanın nominal tek adım değerinin yüzdesi olarak ifade edilmiş halidir. Klasik step motorlarında bu hata % ±1 ile % ± 5 arasında değişmektedir. Sürtünme momenti veya sürtünme kuvveti nedeniyle oluşan konum hataları bu doğrulukla ilgisi olmayan, daha az veya çok olabilen rastgele hatalardır. Ancak her iki tip hata toplanarak sistemin toplam hatası elde edilir. Bu hata değeri kümülatif (birikimli) değildir. Yani rotorun yaptığı her adım ile bu hata miktarı toplanarak gitmez [15].

Çözünürlük: Bir devirdeki adım sayısı olarak tanımlanır. Bu sabit değer, üretim sırasında tespit edilen bir büyüklüktür. Bir step motorunun adım büyüklüğü, çeşitli kontrol düzenleri ile değiştirilebilir. Yarım adım çalışmada adım büyüklüğü normal değerinin (çözünürlüğünün) yarısına indirilir [15].

Etiket geriliminin anlamı: Fren gerilimi olarak da adlandırılan etikette belirtilen gerilim rotor hareketsizken tutma momentinin oluşturulması için gerekli gerilim değeridir [15].

3.3.8. Adım motoru seçimi

Herhangi bir uygulama için adım motoru seçimi yapılacağı zaman, çeşitli faktörlerin göz önünde bulundurulması gerekir. Sistem için hangi tür motor kullanılması gerektiği, motorun içinde kullanılacağı sistemin moment ihtiyacı, kontrolörün karmaşıklığı ve motorun fiziksel karakteristikleri göz önünde bulundurulması gereken faktörlerden birkaç tanesidir. Bunun için adım motorların birbirlerine göre kıyaslanmasında yarar vardır [16].

Değişken relüktanslı adım motoru ile sabit mıknatıslı veya hibrid adım motorlarının karşılaştırılması

Değişken relüktanslı adım motorlarının basit yapıda olma avantajları bulunmaktadır. Bu motorlar rotor üzerinde karmaşık sabit mıknatıslara gerek duymamaktadır. Bu nedenle sabit mıknatıslı adım motorlarına göre daha sağlamdır. Bütün adım motorlarında artan hıza karşı moment düşmektedir. Ancak değişken relüktanslı adım motorları için momentteki bu düşme daha az telaffuz edilir. Uygun bir tasarımla değişken relüktanslı motorlarla 10 000 adım/sn hıza ulaşmak mümkündür. Sabit mıknatıslı ve hibrid adım motorlarında ise nadiren 5 000 adım/sn hıza ulaşılır. Çünkü bu hız seviyesinde bu motorların ürettiği moment çok az ve bu nedenle de sabit mıknatıslı ve hibrid adım motorlarının hızı genellikle 1 000 adım/sn'nin altındadır. Değişken relüktanslı adım motorlarında hıza karşı momentte az düşme meydana geldiği için, diğer adım motorlarının dişli takımına ihtiyaç duyduğu uygulamalarda bu adım motorları kullanılacak olursa, dişli takımı kullanımına gerek kalmamaktadır. Örneğin günümüzde kullanılan çok yeni çamaşır makinelerinde tamburu sürmek için değişken relüktanslı adım motoru kullanılmaktadır [16].

Değişken relüktanslı adım motorlarının dezavantajı da bulunmaktadır. Sinüsoidal uyartım akımlarıyla bile sabit mıknatıslı adım motorları çok sessiz çalışırken değişken relüktanslı adım motorları, tahrik için kullanılan dalga şekli nasıl olursa olsun genellikle gürültülü çalışırlar. Bu nedenle, genellikle gürültünün ve vibrasyonun istenmediği ortamlarda sabit mıknatıslı veya hibrid adım motorları kullanılır [16].

Değişken relüktanslı motorlardan farklı olarak, sabit mıknatıslı ve hibrid adım motorları bir kaynak tarafından beslenmedikleri zaman el ile döndürüldüklerinde dönmeye karşı bir zorluk gösterirler. Bunun sebebi motor enerjili olmadığı halde bu motorlarda sabit mıknatıslar satator kutuplarını çekmektedir. Bu motorlardaki manyetik artık tutma momenti bazı durumlarda arzu edildiği halde bazı durumlarda da kesiksiz hareket istendiği için problem olmaktadır. Uygun kontrol sistemleriyle hem sabit mıknatıslı hem de hibrid adım motorları mikro adımlarla hareket ettirilebilirler.

Genel olarak değişken relüktanslı adım motorları mikro adımlarla hareket ettirilemezler. Değişken relüktanslı adım motorları çoğunlukla tam adımlarla hareket ettirilirler [16].

Bipolar adım motoru ile unipolar adım motorunun karşılaştırılması

Bir unipolar veya bipolar sürücü sistem arasında tercih yapmak, işlem basitliğine, güç/ağırlık oranına bağlıdır. Bipolar adım motorları aynı hacimde unipolar adım motorlarında yaklaşık olarak %30 daha fazla momente sahiptir. Bunun sebebi, herhangi bir anda unipolar motorda sargının sadece yarı bobini enerjilenmektedir. Bipolar motor ise, enerjilendiği zaman tüm sargıyı kullanmaktadır [20].

Bipolar motor ile üretilen yüksek momentin bir maliyeti bulunmaktadır. Bipolar motorlar unipolar motorlara göre daha karmaşık kontrol devreleri gerektirir. Bu da uygulamanın maliyetini etkilemektedir [16].

Sabit mıknatıslı adım motoru ile hibrid adım motorunun karşılaştırılması

Hibrid veya sabit mıknatıslı adım motorları arasında tercih yapmak için, maliyet ve çözünürlük gibi iki önemli hususu dikkate almak gerekir. Bu motorları sürmek için kurulan veya tasarlanan elektronik ve bağlantı opsiyonları genellikle her iki motor tipi için de uygulanabilmektedir [16].

Sabit mıknatıslı adım motorlar şüphesiz imal edilen en ucuz motorlardandır. Bu motorlar bazen teneke yığını olarak bilinirler. Çünkü bu motorların statoru, basılmış metal içerisine yerleşik iki sargıdan ibarettir. Karşılaştırılacak olursa hibrid ve değişken relüktanslı adım motorları yalıtılmış ince sac levhalardan ve sarılması daha zor olan motor sargılarından yapılırlar [16].

Sabit mıknatıslı adım motorları genellikle adım aralığı 30^{o}'den 3.6^{o}'ye kadar değişecek şekilde yapılırlar. Sabit mıknatıslı adım motorunun rotorundaki mıknatıslar rotorun 50 kutuptan fazla olmasına engel olur. Bu nedenle, sabit mıknatıslı adım motorlarında genellikle adım açısının küçük olduğu nadiren görülür. Hibrid motorlarda sabit mıknatıslı rotorun üzerine ve karşılıklı iki ucuna, çok ince ve düzenli bir biçimde dişler açılarak çok kullanılan bir adım büyüklüğü

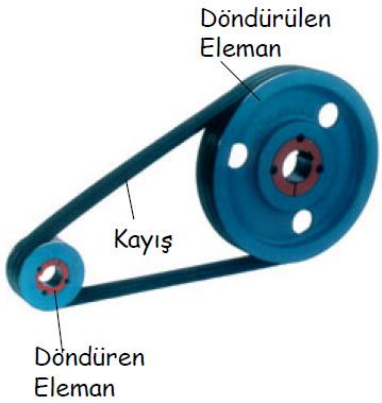
olan 1.8^{o} adım açısı elde edilir. Bu yöntem ile daha küçük adım açıları elde etmek mümkündür [16].

3.4. Güç ve Hareket Aktarım Sistemleri

Elektrik motorlarından elde edilen güç ve hareketin iletilmesini, dönüştürülmesini veya değiştirilmesini sağlayan sistemlerdir. Günümüzde çok çeşitli bileşenleri olması ile birlikte temel mantık olarak birkaç mekanizma kullanılmaktadır.

3.4.1. Kayış kasnak sistemleri

Kayış – kasnak mekanizmaları, millere bağlı dişliler arasında güç iletimi sağlayan mekanizmalardır. Güç iletimi için kayış ile dişliler arasında sürtünmeli temas yani gerginlik olmalıdır. Millere bağlı dişliler veya kasnaklar arasında yarıçap oran farkı varsa güç, hız ve moment dönüşümü elde edilir. Eğer yarıçaplar eşit ise bu dönüşümler olmaz, aktarım birebir gerçekleşir. Kayışlar, çoğunlukla kauçuk veya polimer tabanlı materyallerden üretilirler [20].



Şekil 3.15. Kayış kasnak sistemi

Kayış-kasnak mekanizmalarının avantajlı yönleri aşağıdaki gibi özetlenebilir.

- Yapıları basit, imalatları kolay olup, diğer mekanizmalara göre oldukça ucuz bir yapı oluştururlar.
- Düşük çevre hızları dışında genellikle maliyetleri düşük olan mekanizmalardır.
- Kayış elastik olduğundan aşırı yüklerde titreşim ve darbe söndürücü özellik gösterir. Darbeleri karşılama kabiliyetleri büyüktür.

- Mekanik kayıplar çok düşük olup tipik bir kayış-kasnak mekanizmasında verim %98'e kadar ulaşabilmektedir.
- Eksenleri arasında büyük açıklık bulunan miller arasında basit ve ucuz bağlantı sağlayabilmektedirler.
- Kayış uçlarındaki gürültü önlenebilirse çalışma sessiz olur
- Çok geniş hız ve güç bölgesinde kullanılabilirler.

Bu avantajlarına karşılık aşağıda sıralanan dezavantajlı yönleri de vardır.

- Hareket iletimi sürtünme ile gerçekleştirildiğinden, kayışın kasnak üzerine bastırılması zorunlu olduğundan bir basma kuvvetine ihtiyaç duyulur.
- Basma kuvveti etkisiyle mil ve yataklar daha büyük zorlanmalara maruz kalırlar.
- Kayışta zamanla oluşan gevşeklik nedeniyle mekanizmada gerdirme tertibatına ihtiyaç duyulur.
- Zincir ve dişli çark gibi mekanizmalara nazaran iletilen birim güç başına hacim ve ağırlıkları daha büyüktür.
- Kasnak ile kayış arasında yapıları gereği az da olsa bir kayma mevcuttur. Kısmi kaymalardan dolayı genellikle düz kayış-kasnak mekanizmalarında tam ve sabit çevrim oranı sağlanamaz.

3.4.2. Vidalı mil sistemleri

Vidalı mil, bilya yataklı bir somunun vida dişleri açılmış bir mil üzerindeki sistem sayesinde dönme hareketini doğrusal harekete çeviren makine elemanıdır. Bu hareket esnasında sürtünmenin azaltılabilmesi amacıyla somun ile mil arasında yer alan boşlukta yataklanan bilyalar mevcuttur. Bir iletim hareketi elemanı olan vidalı miller, doğrusal hareketi bu sayede daha az sürtünme ile iletirler. Hassas bir vida olarak yapılan dişli mil ise helezonik yapıdaki kanalları sayesinde bilya yataklarının rahat hareketine olanak tanır. Düşük sürtünme özelliği sayesinde yüksek mekanik verim elde edilir [21].

Sistem, en basit şekliyle somun, vida ve yuvarlanma elemanı olmak üzere üç temel unsurdan oluşur. Vida ile somun arasına uygun bir toleransla yerleştirilmiş

olan çelik bilyalar, sistemin aynı bir rulmanlı yatak gibi çok düşük sürtünme katsayısı ile çalışmasını sağlar. Sistemin tersinir olması ve az boşlukla çalışması da önemli bir özelliğidir [21].

Vidalı milllerin yüksek hassasiyete sahip olmaları, uzun ömürlü olmaları ve düşük kirlilik üretmeleri başlıca özellikleridir. Hassas pozisyonlama ve makine sanayindeki hassas ölçüm sistemlerinde önemli bir yere sahip olan vidalı miller, robotlar gibi çok hassas ekipmanlarda ve daha birçok endüstriyel uygulamalarda kullanılmaktadır.

Şekil 3.16. Vidalı mil kesiti

Vidalı millerin avantajları şunlardır;

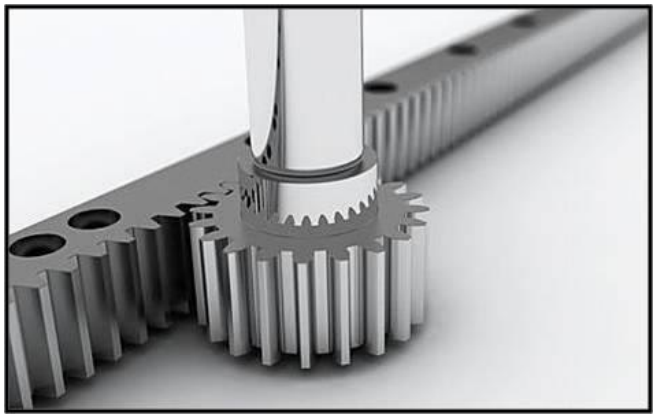
- Doğrusal eğriliği minimum düzeydedir ve çelik olduğu için bu özelliğini korur.
- Üzerinde hareket için bilyeli somun kullanıldığı zaman sürtünme çok çok düşük olur.
- Boşluksuz bilyeli somun kullanılır ise somunda bulunan koruma sayesinde üzerine düşen toz parçacıkları somunun mil üzerinde hareketi sayesinde otomatikman temizlenir dolayısıyla bakım süresi çok uzamış olur ve kirlilikten dolayı sistemde yavaşlamaya neden olmaz.
- Çok hassastırlar.

Vidalı millerin dezavantajları ise şunlardır;

- Pahalıdırlar.
- Yavaştırlar.
- Hata kabul etmezler.

3.4.3. Kramayer dişli sistemleri

Kremayer dişli, şerit şeklinde düz bir dişli olarak düşünülebilir. Kremayer dişli üzerinde yürüyen dişliye ise pinyon dişli denir. Düz dişli ve pinyon dişliden oluşan kremayer dişli sistemi dairesel hareketi lineer (doğrusal) harekete çevirir [22].



Resim 3.3. Kremayer dişle ve pinyon dişli

Kremayer dişli endüstride yaygın olarak kullanılan dişli çeşitlerindendir. Kremayer dişli düz bir yüzeye açılan dişler sayesinde dönebilen dişlinin hareketini bir yönde ilerlemeye çevirir. Kremayer dişlilerin düz ve heliks dişli olmak üzere iki çeşidi vardır.

3.5. Robot Sistemi

Robot, mekanik sistemler ve bunlarla ilişkili kontrol ve algılama sistemleriyle, yazılım algoritmalarına bağlı olarak davranan makinelerdir. Fiziksel algılama yapabilmek için çeşitli sensörler kullanılır. Genel bir yaklaşım ile “Robot, değişik amaçlar için kullanılabilen birçok fonksiyonu olan ve programlanabilen makinelerdir.” seklinde tanımlanabilir [23].

İnsanlar fizikî yapılarından ve güçlerinden dolayı bedensel olarak bütün işleri yapma yeteneğine sahip değillerdir. Bu nedenle gücünün yetmediği yerlerde kullanmak üzere değişik makineler tasarlamışlardır. İlk zamanlarda çok işlevsel olmayan bu ilkel makineler, teknolojinin gelişme sürecine paralel olarak insanlar tarafından geliştirilmiş ve insanın yeteneklerine benzer yetenekleri olan makineler üretilmiştir.

İlk zamanlarda insan kontrolü ile çalışan bu makineler, zamanla daha da geliştirilerek ve çeşitli çevre birimleri kullanılarak insana ihtiyaç duymadan otomatik olarak çalışır hale getirilmişlerdir.

Türkiye de robotik alanında ilk düzenli çalışma 1987 yılında TÜBİTAK (Türkiye Bilimsel ve Teknolojik Araştırma Kurumu) Mühendislik ve Araştırma Grubu ve İTÜ'nün (İstanbul Teknik Üniversitesi) desteği ile "Robot Teknolojisi Araştırma Ünitesi'nin (ROBOTEK)" kurulması ve 1988 yılında Türkiye'nin Birleşmiş Milletler UNDP/UNIDO teşkilatınca düzenlenen "Endüstriyel Robotik Uygulamalar" konulu Avrupa Bölgesel Projesi'ne katılmasıyla başlamıştır [23].

Otomasyon uygulamalarının dışa dönük yüzü endüstriyel robotlardır. Günümüzde robotlar laboratuar uygulamalarından uzay araştırmalarına, endüstriyel alanlarından hizmet sektörüne kadar pek çok alanda kullanılmaktadır. Endüstriyel sektörde kullanılmak üzere tasarlanmış, kaynak yapma, cisim tutma, döküm, yükleme, kalite kontrol ve boyama işlemlerini yapan birçok robot bulunmaktadır.

Robotlar genellikle, üretim maliyetini düşürmek ve yüksek kaliteli üretim için kullanılmaktadırlar. Ayrıca insanın hayatının riske gireceği işlerde ve insanların ulaşamayacağı yerlerde robotlar kullanılmaktadır.

3.5.1. Robot sınıflandırmaları

Endüstriyel robotların sınıflandırılması kumanda sistemlerine, yapılarına vb. değişik yönlerden ele alınarak yapılmıştır. Değişik ülkeler kendilerine göre robotları sınıflandırmıştır. Aşağıda robot konusunda bası çeken Amerika, Japonya ve Fransa'nın kullandıkları robot standartları verilmiştir [23].

Japon endüstriyel robot birliğinin (Japanese Industrial Robot Association – JIRA) robot sınıflandırmasına göre robotlar;

1.Sınıf: Elle Çalıştırılan Robotlar (Manual-Handling Device): Bu robotlar birçok derece de özgürce çalışabilmektedir. Fakat operatör tarafından kullanılması gerekir.

2. Sınıf: Sabit Dizi Robot (Fixed-Sequence Robot): Bu robotlar önceden belirlenmiş sabit isleri yapar ve modifiye edilemeyen robotlardır.

3. Sınıf: Değişken Dizi Robot (Variable-Sequence Robot): Sınıf 2 ile aynıdır. Fakat modifiye edilebilir.

4. Sınıf: Playback Robot :Bu robotlarda operatör robota ilk başta hangi hareketleri yapması gerektiğini robota gösterir. Robot bu hareketleri kaydeder ve daha sonra kendisi aynı hareketleri yapar.

5. Sınıf: Sayısal Kontrol (Numerical Control Robot): Kullanıcı robota yapacağı işleri bir program aracığıyla bildirir. Robot yüklenen programa göre iş yapar. Programın dışında herhangi bir iş yapma veya karar verme yetisi yoktur.

6. Sınıf: Akıllı Robot (Intelligent Robot): Bu robotlar çevreyi algılar ve çevresindeki şartların değişimlerine göre kendisini yeniden konumlandırarak görevini en iyi şekilde yerine getirir.

Amerika robotik enstitüsü'nün (The robotics institue of America – RIA) robot sınıflandırması incelenecek olursa Japonya ile benzer sınıflandırmayı kullandığı görülür. Fakat Japonya'nın sınıflandırmasındaki Sınıf 3 ve Sınıf 6 arası robotları göz önüne almıştır.

Fransız Sanayi Robotları Birliği'nin (The Association Française de Robotique – AFR) robot sınıflandırmasına göre;

Tip A: Elle çalıştırılan robotlar. JIRA sınıflandırılmasındaki 1. Sınıf robotları ile aynı kategoridedir.

Tip B: Otomatik olarak elle çalıştırılan robotlar. Önceden belirlenen işlemleri otomatik olarak yapar. JIRA sınıflandırmasındaki 3.sınıf robotlara eşdeğerdir.

Tip C: Programlanabilir, servo kontrollü, devamlı veya noktadan-noktaya yörüngeli robotları içerir. JIRA'nın 4.sınıftaki robotuna eşdeğerdir.

Tip D: Tip C ile aynıdır. Fakat çevresindeki bilgilere göre hareket etme kabiliyetine sahip robotlardır. JIRA sınıflandırmasındaki 6.sınıf ile aynıdır.

3.5.2. Robotların avantaj ve dezavantajları

Robotların hayatımıza kattığı avantajlar tartışmasız çok fazladır. Bunun yanında her yeni teknolojinin yararlarının yanında zararlarının da olduğu bir gerçektir. Aşağıda robotların avantajları ve bu avantajlarının yanında getirdiği dezavantajları incelenmiştir [23].

Avantajları;

- Robotlar ve otomasyon birçok alanda üretilen ürünlerdeki ve verilen hizmetlerdeki üretkenliği, güvenliği, verimliliği, kaliteyi ve tutarlılığı artırmıştır.
- Robotlar tehlikeli ve zararlı çevre şartlarında, hayat desteği, konfor ve güvenlik gerektirmeyen durumlarda rahatça çalışabilir.
- Robotlar çoğu durumda, ışıklandırma, klima, havalandırma ve gürültü koruması gibi rahat çevre şartlarına ihtiyaç duymazlar.
- Robotlar yorulmadan, can sıkıntısı olmadan, başı ağrımadan, sağlık sigortası ve tatil gerektirmeden devamlı olarak çalışabilirler.
- Robotlara dışarıdan bir müdahale olmadığı zaman, birçok kez aynı doğrulukta çalışabilirler.
- Robotlar insanlara göre daha yanlışsız çalışırlar. Tipik olarak doğrulukları bir inçin (2,54 cm) birkaç bini kadardır. Yeni silikon entegre yongalı ele sahip robotlar mikro inç düzeyinde doğruluğa sahiptir.
- Robotlar birçok uyarıcılı işleri aynı zamanda yapabilirler.

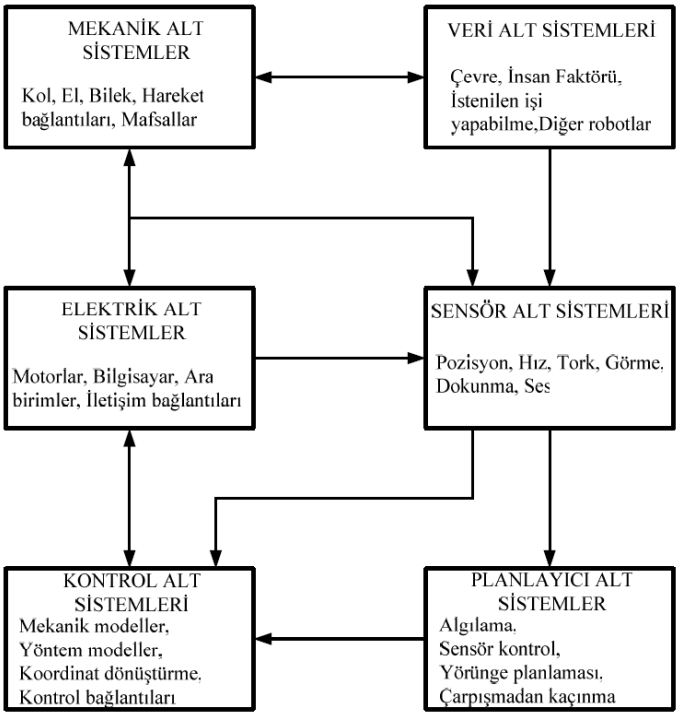
Dezavantajları;

- Robotlar insan işçilerin yerini alarak ekonomik problemler meydana getirirler. Bunlar, düşük maaş, sosyal problemler, işsizlik, işçilerde memnuniyetsizlik ve kızgınlık gibi sıralanabilinir.
- Robotların küçük hataları insanlara zarar verebilir. Bundan dolayı çalışanlar veya robot operatörleri sağlık yönünden zarar görebilir.

- Robotların insanlara göre; serbest dönme derecesi, beceri, algılama, görme, gerçek zamanlı cevap gibi davranışlarda kapasiteleri sınırlıdır.
- Robot sistemleri ve bileşenleri pahalıdır. Robot sistemi için birçok işlem ve mekanizma gereklidir.

3.5.3. Robot sistemi bileşenleri

Robot sistemlerde kullanılan elemanlar, sistemin çok önemli görevlerinden birini yaptıklarından dolayı, her bir eleman sistem içerisinde çok büyük bir önem taşımaktadır. Robot sistemleri genel olarak mekanik ve elektronik sistem elemanlarının uygun bir biçimde bir araya getirilmeleri ile meydana gelir. Robot sistemi alt parçaları; kontrol birimi, elektrik, sensör, mekanik, planlama, yapım ve yazılım alt bölümlerimde incelenebilinir. Şekil 3.17'de robot alt sistemleri görülmektedir [23].



Şekil 3.17. Robot alt sistemleri [23]

Robot sistemlerinin çalıştırılabilmesi için gereken temel sistem bileşenleri;

- Robotun gövdesi
- Robot uç elemanı

- Kontrol ve programlama birimi
- Güç sistemi ve bağlantı elemanları

3.5.4. Manipülatörün yapısına göre robotların sınıflandırılması

Manipülatör (gövde tipi) yapısına göre sınıflandırmada robotlar eksenlerinin mekanik yapısı itibariyle değişik koordinat sistemlerine göre sınıflandırırlar. Bu tarz robotlardaki eksenlerin tiplerine göre değişik özellikleri vardır. Bu özellikler prizmatik eklemler (prismatic), aşağı-yukarı dönüşlü (revolute) eklemler, küresel (spherical) eklemler diye sınıflandırılır. Manipülatör yapısına göre sınıflandırmada her eklem biçimi bu özelliklerden birini veya birkaçını içerir. Bu gösterimde P harfi prizmatik, R harfi aşağı-yukarı döner özelliği belirtir. Örneğin bir robot 3 prizmatik ve 3 aşağı-yukarı dönüşlü eksene sahipse 3P3R olarak gösterilir [23].

Aşağıda manipülatör sınıflandırmasına göre robot çeşitleri ve sahip olduğu eksen özellikleri verilmiştir [23].

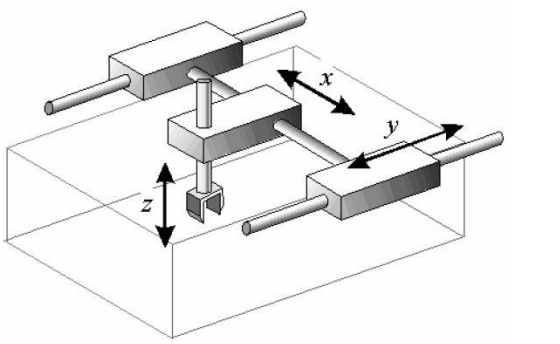
- Kartezyen Koordinatlı Robotlar (Üç prizmatik-kayar- eksenli, 3P)
- Silindirik Koordinatlı Robotlar (İki prizmatik ve bir döner eksenli, R2P)
- Küresel Koordinatlı Robotlar (Bir prizmatik ve iki döner eksenli, 2RP)
- Eklemli Robotlar (Üç aşağı-yukarı döner eksenli, 3R)

Kartezyen koordinatlı robotlar

Kartezyen koordinatlı robotun ana gövdesi, dik açılı üç tane prizmatik-kayar-eksenden oluşur. Bu nedenle önceden açıklandığı gibi 3P ile belirtilir. X,Y,Z, eksenlerinde doğrusal olarak hareket etme yeteneğine sahip bu robot tipi sadece tutma ve taşıma yeteneğine sahiptirler. Basit bir yapıya sahip olduklarından dolayı hareketlerin planlanması basittir. Bu cins robotlarda; pozisyon hesaplamaları, robot uç elemanının bulunduğu pozisyon, mafsalların o anda olduğu yerde bulunduğundan dolayı çok basittir [23].

Şekil 3.18'de görüldüğü gibi çalışma alanları robotun yapısından daha küçüktür. Dönme, eğilme ve bükülme işlemlerini gerçekleştiremezler. Çalışma alanları

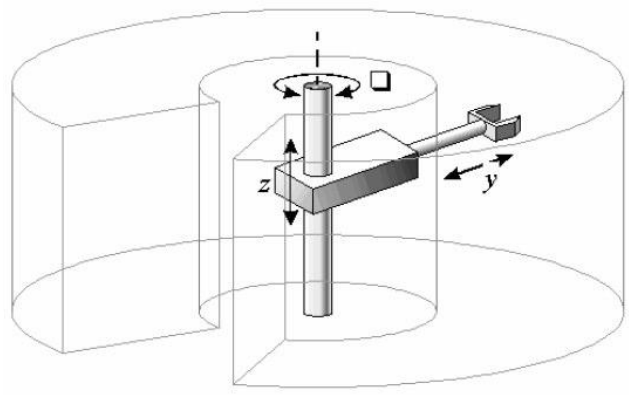
dikdörtgen veya kare prizma şeklindedir. Diğer robot türlerine göre yük taşıma kapasitesi daha büyüktür. İnsan gücünün taşıma kapasitesinden fazla olan yüklerin taşınmasında kullanılır. Bu nedenle genellikle yükleme ve boşaltma işlerinde, fabrikalar da ağır yükleri taşımak amacı ile fabrikaların tavanlarına monte edilerek kullanımı yaygındır. Rutubetli ve ıslak ortamlarında kullanılabilirler [23].



Şekil 3.18. Kartezyen koordinatlı robot ve çalışma alanı

Silindirik koordinatlı robotlar

Silindirik koordinatlı robotlarda Şekil 3.19'da görüldüğü gibi düşey hareketleri gerçekleştiren prizmatik mafsal yerine, kendi etrafında dönebilen mafsal kullanılmıştır. Bu mafsal düşey eksende kendi etrafında dönebildiği gibi, aynı zamanda da diğer prizmatik mafsalları dönen mafsala dik olacak şekilde üzerinde taşımaktadır [23].



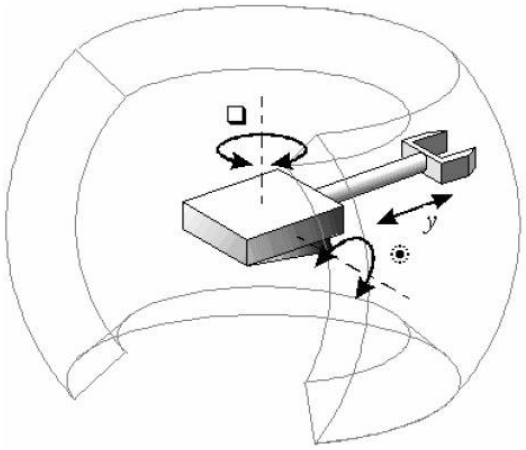
Şekil 3.19. Silindirik koordinatlı robot ve çalışma alanı [23]

İki prizmatik ve bir döner eksenden oluştuğundan R2P ile gösterilir. Silindirik koordinatlı robotun hareket sahası dairesel bir silindire benzer.

Silindirik koordinatlı robotların çalışma alanları, prizmatik kolun maksimum uzanma boyu ve minimum uzanma boyu ile dönen mafsalın bu sınırlar içerisinde taradığı alanın düşey düzlemdeki maksimum uzanma mesafesi ile sınırladığı silindirik alanıdır [23].

Küresel koordinatlı robotlar

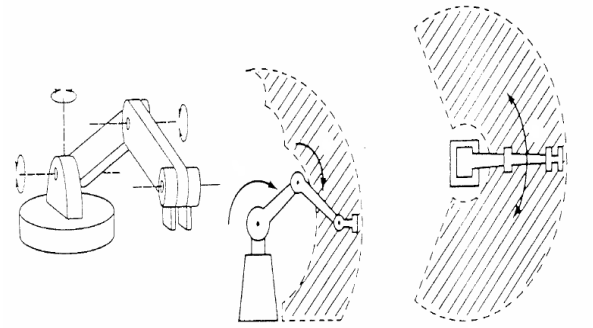
Küresel koordinatlı robot, kaldırma eklemli dönen bir taban ve teleskopik bir koldan oluşur. Kolun hareketleri içe ve dışa doğrudur. Bu robotun yapısı askeri bir tankın kule kısmına benzer. Bir prizmatik ve iki döner eksenden oluştuğundan dolayı 2RP ile gösterilir. Şekil 3.20'de küresel koordinatlı bir robot ve hareket sahası görülmektedir [23].



Şekil 3.20. Küresel koordinatlı robot ve çalışma alanı

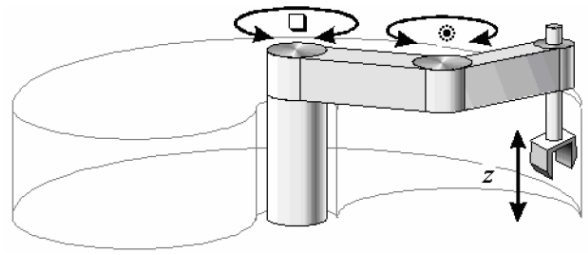
Eklemli robotlar

Eklemli robotlar, dairesel hareket edebilen bel (waist), bir omuz (should) ve yine dairesel hareket eden dirsek (elbow) mafsallarından oluşur. İş yapma yeteneklerine göre diğer robot türlerine nazaran insan hareketlerine daha benzer biçimde hareket yapabilirler. Üç döner eksenden oluştuğundan dolayı 3R ile gösterilir. Şekil 3.21'de eklemli bir robot ve hareket sahası görülmektedir [23].



Şekil 3.21. Eklemli robot ve hareket alanı [23]

Çok dikkat gerektiren tasıma ve montaj islerini daha hassas olarak yapabilmek amacıyla basit geometrili kollara sahip olan eklemli robot sınıfında SCARA (Selective Compliance Automatic Robot Arm – Seçici Uyumlu Montaj Robot Kolu) tipi robot geliştirilmiştir. Şekil 3.22'de SCARA tipi eklemli bir robot şekli görülmektedir [23].



Şekil 3.22. SCARA robot kolu ve hareket alanı [23]

SCARA tipi robotların çok hızlı hareket yeteneği ve yüksek doğrulukta çalışması endüstride toplama, yerleştirme ve parça ekleme gibi montaj işlerinde kullanım oranını arttırmıştır. SCARA robot türü bütün robot manipülatörlerinin özelliklerini üzerinde toplamıştır. İki tane dönebilen mafsal aracılığı ile yatay düzlemde koordinat ayarlaması yaparken, bir tane prizmatik mafsal aracılığı ile düşey düzlemle olan ilişkiyi sağlamaktadır. Düşey düzlemde hareket eden kolun uzanabilme yeteneği olması ve bu kola robot uç elemanı eklemek suretiyle, en karmaşık montaj işlerinde kullanılan robot durumuna gelmektedir [23].

Manipülatör yapısına göre robot çeşitlerinin karşılaştırılması

Çizelge 3.1'de tüm robot çeşitlerinin tip, eksen özellikleri, kullanım alanları ve kullanım sonuçlarına göre karsılaştırılması görülmektedir.

Çizelge'de görüldüğü gibi robot türleri içinde eklemli robotların kullanım alanları endüstride daha fazla yer almaktadır. Buna karşın yapıları karmaşık ve hareket analizleri diğer türlere göre daha zordur. Endüstride kullanım yeri, amacı, ortam şartları ve maliyet göz önüne alınarak üretime uygun robot çeşidi seçilmelidir [23].

Çizelge 3.1. Manipülatör yapısına göre robot çeşitlerinin karşılaştırılması [23]

Robot Tipi	Eklem Tipleri	Kullanım Alanları	Kullanım Sonuçları
Kartezyen Robotlar	Prizmatik bel Prizmatik omuz Prizmatik dirsek	Demiryolu köprü inşaatları Büyük makine montaj hatları	Kinematik modelleri basittir. Rijit bir gövdeye sahiptir. Hareket analizleri basittir. Çalışması için büyük alanlar gerekir. Büyüklüğüne göre iş alanı küçüktür.
Silindirik Robotlar	Döenbilen bel Prizmatik omuz Prizmatik dirsek	Büyük makine montaj sanayi Basit montaj-demontaj hatları	Kinematik modelleri basittir. Hareket analizleri basittir. Güçlü hidrolik elemanlar kullanılır. İş alanları sınırlıdır. Tozlu ve ıslak alanlarda çalışmaları zordur.
Küresel Robotlar	Döenbilen bel Dönebilen omuz Prizmatik dirsek	Montaj sanayi Nükleer sanraller	Büyük alanlara uzanabilirler. Zeminden uzaktaki nesneleri tutabilirler. Kinematik modelleri karışıktır. Hareket analizi zordur.
Eklemli Robotlar	Döenbilen bel Dönebilen omuz Dönebilen dirsek	Otomobil montaj sanayi, Otomobil boya sanayi, Elektronik montaj sanayi, Nükleer santraller, Tıbbi araç-gereç yapım sanayi	Maksimum esnekliğe sahiptirler. İş alanı robot büklüğü ile orantılıdır. Elektrik motorları kullanılabilir. Cisimleri altlarından tutabilir. Kinematik yapıları karmaşıktır. Hareket analizi zordur. Kolların rijit ayarı zordur.

3.5.5. Kontrol döngüsü tipine göre sınıflandırma

Kontrol sistemi, robotların yapacağı işlemlerdeki giriş değişkenlerine bağlı olarak hareketi ve görevini gerçekleştirir veya giriş değişkenlerinin durumunu göz önüne almadan hareketini ve görevini gerçekleştirir. Bu durumlara göre kontrol tipine göre robotlar 2'ye ayrılır [23].

- Açık Döngü Kontrol Sistemi
- Kapalı Döngü Kontrol Sistemi

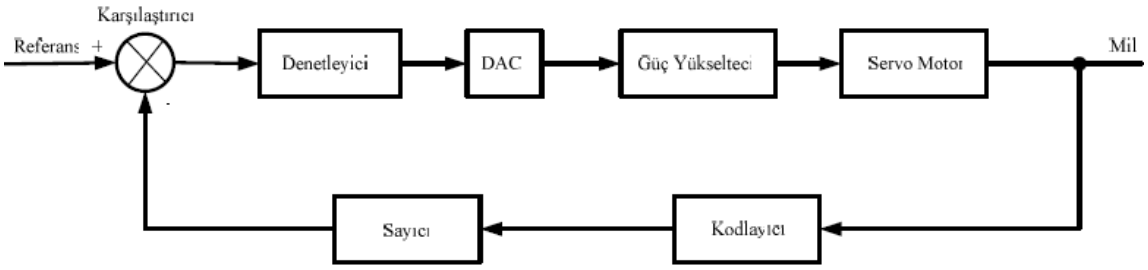
Açık döngü kontrol sistemi

Açık döngü kontrol sistemlerinde, çıkışın giriş üzerinde herhangi bir etkisi yoktur. Yani giriş komutuna göre oluşan çıkış durumundan girişe geri besleme yapılmaz [23].

Kapalı döngü kontrol sistemi

Kapalı döngü kontrol sistemlerinde, açık döngü kontrol sistemlerinin tam tersine çıkışın giriş üzerinde etkisi vardır. Çıkışta meydana gelen değişimler robotun bir sonraki hareketinin durumunu değiştirmektedir [23].

Kapalı döngü kontrol sistemine örnek olarak tek bir eksenin hareketi için servo motor kullanarak yapılan bir sistemin blok seması Şekil 3.23'de görülmektedir [23].



Şekil 3.23. Kapalı döngü kontrol sistemine örnek sistem [23]

3.5.6. Robot tahrik sistemleri

Robotların hareketini sağlayan düzenler tahrik sistemleri (actuators) olarak isimlendirilir. Tahrik düzenleri robotların kasları olarak bilinir. Eklemler robotların iskeleti olarak düşünüldüğünde, robotun hareket etmesini, sağa-sola dönmesini sağlayan kas görevini gören sistemler tahrik sistemleridir. Tahrik düzenekleri eklemlerin hızını artırmak, yavaşlatmak, yükü kaldırmak için ekonomik, doğru, güvenilir biçimde yeterli güce sahip olmalıdır. Birçok tipte tahrik sistemleri mevcuttur. Aşağıda bu tahrik sistemleri verilmiştir [23].

- Hidrolik tahrik sistemleri
- Pnömatik tahrik sistemleri
- Elektrik motorları

- DC motorlar
- Step motorlar
- Servo motorlar

Hidrolik tahrik sistemleri

Hidrolik tahrik sistemleri akışkanların hareket ettirilmesi prensibine göre çalışırlar. Hidrolik sistemlerde genellikle basınçlı yağ kullanılır. Hidrolik sistemlerde güç-ağırlık oranı büyüktür. Tepki hızları düşüktür, fakat sağladıkları kuvvet büyüktür. Özellikle otomobil üretimi gibi ciddi güç gerektiren alanlarda tercih edilen büyük robotlar hidrolik tahrik sistemini kullanırlar.

Pnömatik Tahrik Sistemleri

Pnömatik tahrik sistemlerinin çalışma prensibi hidrolik sistemlere çok benzemektedir. Bu sistemlerde akışkan olarak basınçlı hava kullanılır. Pnömatik tahrik sistemleri hidrolik sistemlere göre daha düşük güçtedirler, fakat daha ucuzdurlar. Bu sistemlerde elektrik kontrollü selenoid vana ile basınçlı hava silindiri döndürür. Gazların sıvılara göre sıkıştırılabilmesi göz önüne alındığında, pnömatik sistemlerin kontrol uyarılarına verdikleri tepki hidrolik sisteme göre yaklaşık 4 kat daha yavaştır. Pnömatik sistemlerin ana problemini havanın belli miktarda sıkıştırılabilmesi oluşturmaktadır. Bu durumdan dolayı yük altında hava sıkışarak istenen sonucu tam verememektedir [23].

Elektrik motorlu tahrik sistemleri

Robotik alanında en fazla kullanılan tahrik sistemi elektrik motorları ile yapılan sistemdir. Bunun nedeni, elektrik motorlarının yeteneklerinin fazla olması, diğer sistemlere göre bakım ihtiyacının az ve boyutlarının küçük olmasıdır [23].

Robot sistemlerinde tahrik elemanı olarak kullanılan elektrik motorları; DC motorlar, servo motorlar ve adım motorlar olarak sınıflandırılabilir. Bu tahrik elemanları üçüncü bölümde incelendiği için burada açıklanmayacaktır.

3.6. Plaka Tanıma

Teknolojinin ilerlemesiyle birlikte, rutin olarak yapılan bazı işlemlerr artık bilgisayar kontrollü sistemlere yaptırılmaktadır. Çünkü bu sistemler yorulmaz, sıkılmaz, hastalanmaz, acıkmaz, sosyal ihtiyaçları yoktur ve öğretilenler doğrultusunda neredeyse sıfır hata ile iş yaparlar. Araç sayısındaki artışa bağlı olarak, trafik akışının ve güvenliğinin kontrolünde de otomasyon sistemlerine yönelim artmıştır. Hız kontrolü, araç sayımı, araç tanımlaması, kural ihlalleri, plaka okuma gibi birçok alanda otomasyon sistemlerinden yararlanılmaktadır.

Günümüzde trafik denetimi için, radyo frekanslarını kullanan radarlar, mikrodalga detektörleri, yolun altına yerleştirilen tüpler ve loop (döngü) detektörleri bulunmaktadır. Ancak bu donanımların kurulumu ve algılayıcıların fazlalığı bu sistemleri pahalı hale getirmektedir. Bununla beraber, bu sistemlerin işletimi de zordur [24].

Resim işleme; dijital ortama alınan resmin, bilgisayar ve yazılım yardımıyla amaca uygun şekilde değiştirilerek kullanılması veya alınan resmin işlenerek üzerinden gerekli bilgilerin alınmasıdır. Bu sayede elektronik ortamdaki resimlerden, istenilen görüntü sonuçları ve veriler elde edilir. Resim işleme günümüzde birçok alanda kullanılmaktadır. Resim işlemenin faydaları; kontrol işlemleri için insan gücünün kullanımını azaltarak, bunun yerine 24 saat aralıksız çalışabilen ve de bir insana göre çok daha hızlı işlem yapabilen makinelere işleri devrederek, maddi yönden ve performans yönünden büyük kazanç sağlamasıdır. Fakat bir makine bir insan kadar zeki olamaz ve sağduyulu davranamaz, sadece kendisine öğretilen işlemleri çok hızlı bir şekilde yerine getirir [25].

Plaka tanıma sistemi (PTS), elde edilen araç görüntüsünün bilgisayar ortamında işlenerek resim üzerinden plaka bilgilerinin elde edilmesidir. Çünkü her aracın plakası kendisine özeldir ve araca ait tüm bilgilere plaka bilgisi ile rahatlıkla ulaşılabilir.

Plaka tanıma sistemleri; tesis ya da bina otoparklarının otomatik girişlerinde, ücretli otoparklarda, ücretli geçiş sistemlerinde, ülkelerin sınır kontrollerinde, polis

tarafından aranan araçların bulunmasında ve kural ihlali yapan araçların tespiti gibi birçok alanda kullanılabilmektedir.

Bir aracın plaka bilgilerinin resim üzerinden tanımlanması genel olarak üç aşamadan oluşur:

1- Görüntüde plaka bölgesinin tespit edilmesi,
2- Tespit edilen plaka bölgesindeki karakterlerin ayrıştırılması
3- Ayrıştırılan karakterlerin tanımlanması

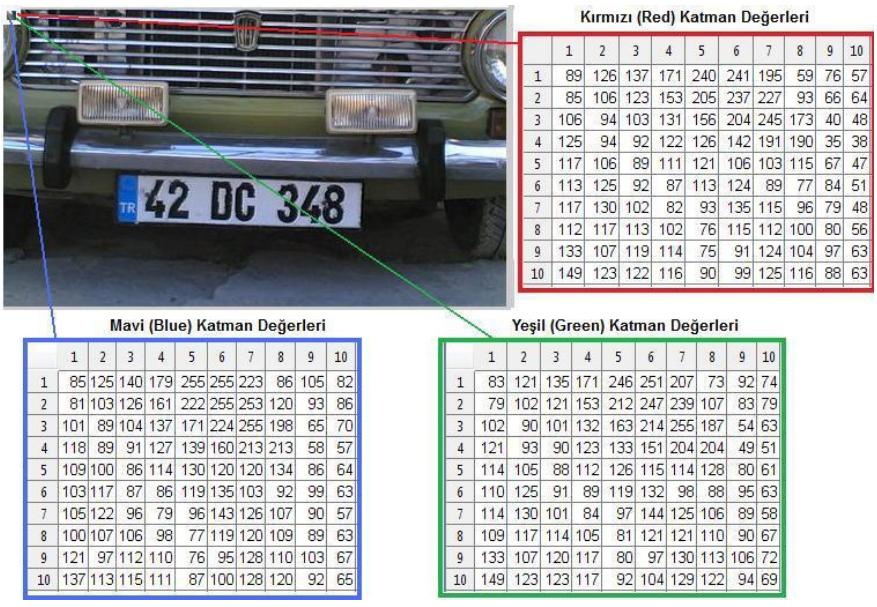
Plaka tanımlamak için çekilen resimlerde, tanımlamayı zorlaştıracak birçok etmen vardır. Bunlardan bazıları; ışık şiddeti, parlaklık, şekilsel bozukluklar (plakanın eğimli olması), plaka üzerindeki kir, pas, çamur, vida, özel bilgiler (ülke kodu vb) olabilir. Daha önce de bahsedildiği gibi otomatik plaka tanımlama sistemleri, dijitalleştirilmiş görüntünün işlenmesidir. Dijital görüntünün işlenmesi aşamalarında, bahsedilen bozucu etmenlerin (gürültü) sonuca etkisini önlemek veya en aza indirgemek amacı ile bazı teknikler kullanılmaktadır. Görüntü işlemede kullanılan teknikler ana başlıklar altında aşağıda verilmiştir [26].

- Aritmetik İşlemler (toplama, çıkarma, çarpma, bölme, lojik and, nand, or, nor, xor, xnor)
- Nokta İşlemleri (eşikleme, uyarlanabilir eşik, kontrast yayılımı, histogram eşitleme, logaritmik operatörler)
- Geometrik İşlemler (ölçekleme, döndürme, yansıtma, öteleme)
- Görüntü Analizi (yoğunluk histogramı, sınıflandırma, bağlantılı parçaların belirlenmesi)
- Morfolojik İşlemler (aşındırma, genişletme, açma, kapama, inceltme, kalınlaştırma, iskelet çıkarma)
- Dijital Filtreler (ortalama filtre, medyan filtre, Gauss ve Laplace filtreleri, frekans filtreleri, bulanık filtre)
- Özellik Dedektörleri (Robert Cross, Sobel, Canny kenar dedektörleri, çizgi dedektörü)
- Görüntü Dönüştürücüler (uzaklık dönüşümü, Fourier dönüşümü, Hough dönüşümü)

Plaka tanıma sistemini tasarlayan kişi amacına ve yöntemine uygun olarak yukarıdaki tekniklerden bazılarını kullanabilir. Bu tekniklerin tamamını burada incelemek mümkün değildir. Fakat önemli görülen bazıları hakkında bilgi verilecektir.

3.6.1. Renkli görüntü

Gerçek ve esas olan görüntü renkli görüntüdür ve burada milyonlarca tonda farklı renk vardır. Görüntüde her bir piksel için R (Kırmızı), G (Yeşil), B (Mavi) olarak kodlanmış ayrı değerler tutulur. Her pikseli temsil etmek için kullanılan bitlerin sayısı piksel derinliği olarak adlandırılır. Piksel derinliği 24 bit olan bir resimde, her R (Kırmızı), G (Yeşil) ve B (Mavi) için 8 bit ayrılmıştır ve bunların her biri 0 ile 255 arasında değişen değerler alırlar. Bir piksel için kullanılabilecek renk sayısı 255x255x255= 16 777 216'dır. Kırmızı, yeşil ve mavi ana renk bileşenlerinin her biri ayrı ayrı 3 farklı matriste tutulur. Bu üç matrisin bir arada, üst üste görüntülenmesi ile gerçek renk bileşenleri meydana gelir. Renkli görüntüye bir örnek Şekil 3.24'de gösterilmiştir [25].



	1	2	3	4	5	6	7	8	9	10
1	89	126	137	171	240	241	195	59	76	57
2	85	106	123	153	205	237	227	93	66	64
3	106	94	103	131	156	204	245	173	40	48
4	125	94	92	122	126	142	191	190	35	38
5	117	106	89	111	121	106	103	115	67	47
6	113	125	92	87	113	124	89	77	84	51
7	117	130	102	82	93	135	115	96	79	48
8	112	117	113	102	76	115	112	100	80	56
9	133	107	119	114	75	91	124	104	97	63
10	149	123	122	116	90	99	125	116	88	63

	1	2	3	4	5	6	7	8	9	10
1	85	125	140	179	255	255	223	86	105	82
2	81	103	126	161	222	255	253	120	93	86
3	101	89	104	137	171	224	255	198	65	70
4	118	89	91	127	139	160	213	213	58	57
5	109	100	86	114	130	120	120	134	86	64
6	103	117	87	86	119	135	103	92	99	63
7	105	122	96	79	96	143	126	107	90	57
8	100	107	106	98	77	119	120	109	89	63
9	121	97	112	110	76	95	128	110	103	67
10	137	113	115	111	87	100	128	120	92	65

	1	2	3	4	5	6	7	8	9	10
1	83	121	135	171	246	251	207	73	92	74
2	79	102	121	153	212	247	239	107	83	79
3	102	90	101	132	163	214	255	187	54	63
4	121	93	90	123	133	151	204	204	49	51
5	114	105	88	112	126	115	114	128	80	61
6	110	125	91	89	119	132	98	88	95	63
7	114	130	101	84	97	144	125	106	89	58
8	109	117	114	105	81	121	121	110	90	67
9	133	107	120	117	80	97	130	113	106	72
10	149	123	123	117	92	104	129	122	94	69

Şekil 3.24. Renkli resimden görüntü ve resmin ilk 10x10 piksellik alanının RGB katmanındaki renk bilgileri [25]

Gerçek renk bileşenlerine sahip bir görüntü üzerinde işlem yapmak oldukça zordur. Çünkü her piksel üzerinde 24 bitlik bir veri vardır ve bunlarla ilgili matematiksel işlemler yapmak çok fazla zaman kaybına sebep olur.

Bu yüzden görüntüleri işlemek için resimleri gri seviyeye hatta daha hızlı işlem yapabilmek için siyah beyaz resim formatına dönüştürmek gerekir. Her pikselde sadece 1 veya 0 bilgisini alarak işlem yapmak çok daha hızlı olur. Yine renkli bir resmin, siyah beyaz bir resme göre bilgisayar belleği üzerinde kaplayacağı alan çok fazladır. Bu veriler incelenmek için sürekli işlemciye gönderilip, buradan işlenmiş verinin tekrar alınacağı da düşünülürse, siyah beyaz görüntü oldukça avantajlı duruma gelir. Fakat siyah-beyaz resim ile renkli resim arasından çok büyük detay farkı vardır. Bu yüzden yapılacak işleme göre resmin dönüştürüleceği format iyi seçilmelidir [25].

3.6.2. Renkli resmin gri seviyeli resme dönüşümü

Gri seviye bir resim ile renkli bir resim arasındaki fark, renkli resmin bir pikselinde 3 farklı renk bilgisi yani 3 farklı değer tutulurken, gri seviyede sadece tek bir değer tutulmasıdır. Gri seviyeli resme dönüşüm için bu 3 farklı 0 ile 255 arasında değişen renk bilgisi değerlerini, tek bir 0 ile 255 arasında değişen değer haline getirmek gerekir.

Bunu için en basit yol bu 3 değerin toplanarak, 3'e bölünmesidir. Bu sayede 3 farklı değer tek değer haline getirilmiş olur. Renkli bir resmin Eş. 3.1 kullanılarak gri seviyeli görüntüye çevrilmiş hali Resim 3.4'de gösterilmiştir [25].

$$Y = (R\ (Kırmızı) + G\ (Yeşil) + B\ (Mavi))/3 \quad (3.1)$$

Resim 3.4. Renkli resim ve (R+G+B)/3 formülüyle gri seviyeli görüntüye çevrilmiş resim [25]

Yapılan deneyler sonucunda insan gözünün renklere duyarlılığı incelenerek, gözün renklere göre duyarlılık eğrisi çıkarılmıştır. Bu eğri çıkarılırken beyaz ışığa olan göz duyarlılığı 1(bir) olarak alınmıştır. Eğri incelenerek insan gözünün en iyi yeşil rengi algıladığı görülmüştür. Yeşil renk aynı zamanda görülebilen ışık dalga boylarının orta kısmında yer almaktadır. Yeşil renkten sonra kırmızı, daha sonra da mavi renk en iyi algılamaktadır. Gözün yeşil, kırmızı ve maviyi algıladığı noktalardaki genlik değerleri toplamı 1'i , yani beyaz rengi verir [25].

0,587 yeşil (G) + 0,299 kırmızı (R) + 0,114 mavi (B) =1 beyaz (3.2)

Eş. 3.2'den anlaşıldığı gibi üç ana renk, belirli oranlarda karıştırılarak beyaz renk elde edilir. Eş. 3.3 kullanılarak gri seviyeli görüntüye çevrilme işlemi Resim 3.5'de görülmektedir.

Y=0.299R (Kırmızı)+0.587G (Yeşil)+0.114B (Mavi) (3.3)

Resim 3.5. Renkli resim ve Y=0.299R+0.587G+0.114B formülüyle gri seviyeli görüntüye çevrilmiş resim [25]

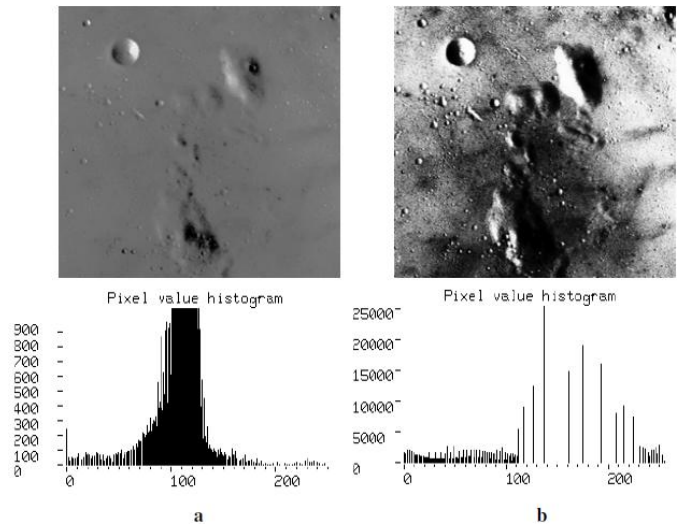
3.6.3. Histogram ve histogram eşitleme

Günün farklı zamanlarında çekilen resimler farklı gri parlaklık/kontrast seviyelerine sahiptir. İnsanlar olarak bizler ortamdaki küçük ışık değişimlerinden pek etkilenmeyiz. Örneğin bir arkadaşımızın yüzünü öğle güneşi altında da, florasan ışığında da, aksam güneşinde de tanıyabiliriz. Ama bilgisayarlar için ortamdaki ışık değişikleri sorun olabilmektedir. Örneğin bir ağaç görüntüsünün 0..255 aralığındaki gri seviyelerinde kaydediliyor olduğunu farz edelim. 0 tam siyahı, 255 tam beyazı, aradaki değerler de gri tonlarını gösteriyor olsun. Bu görüntüyü öğlen vakti kaydettiğimizde elimizdeki verilerin ortalama değeri 255'e yakın değerler olurken,

aynı ağacın görüntüsünü aksam kaydettiğimizde ortalama değer 0'a daha yakın olacaktır ve bu bilgisayar ile tanıma açısından sorun oluşturur. Parlaklık ve kontrast seviyelerindeki bu dengesizlik sebebiyle oluşan problemleri aşabilmek için, resmin gri ton dağılımının homojen yapılandırılması sağlanmalıdır [27].

Histogram, sayısal bir resim içerisinde her renk değerinden kaç adet olduğunu gösteren grafiktir. Bu grafiğe bakılarak resmin parlaklık durumu ya da tonları hakkında bilgi sahibi olunabilir [26].

Bu yöntem histogramı dar olan resimler ya da resim içindeki bölgeler için daha iyi sonuç verir. Önce resmin histogramı bulunur. Histogramdan yararlanılarak kümülatif histogram bulunur. Kümülâtif histogram, histogramın her değerinin kendisinden öncekiler ve kendisinin toplamı ile elde edilen değerleri içeren grafiktir. Kümülâtif histogram değerleri yeni resimde olmasını istediğimiz maksimum renk değerleri ile çarpılıp resimdeki toplam nokta sayısına bölünerek normalize edilir. Daha sonra normalize olmuş histogram değerleri ile resmin renk değerlerini tekrar güncellenirse o resme histogram eşitleme metodu uygulanmış olur. Şekil 3.25'de histogram eşitleme öncesi ve sonrası resimlerle bunların histogramlarının nasıl değiştiği görülmektedir [28].

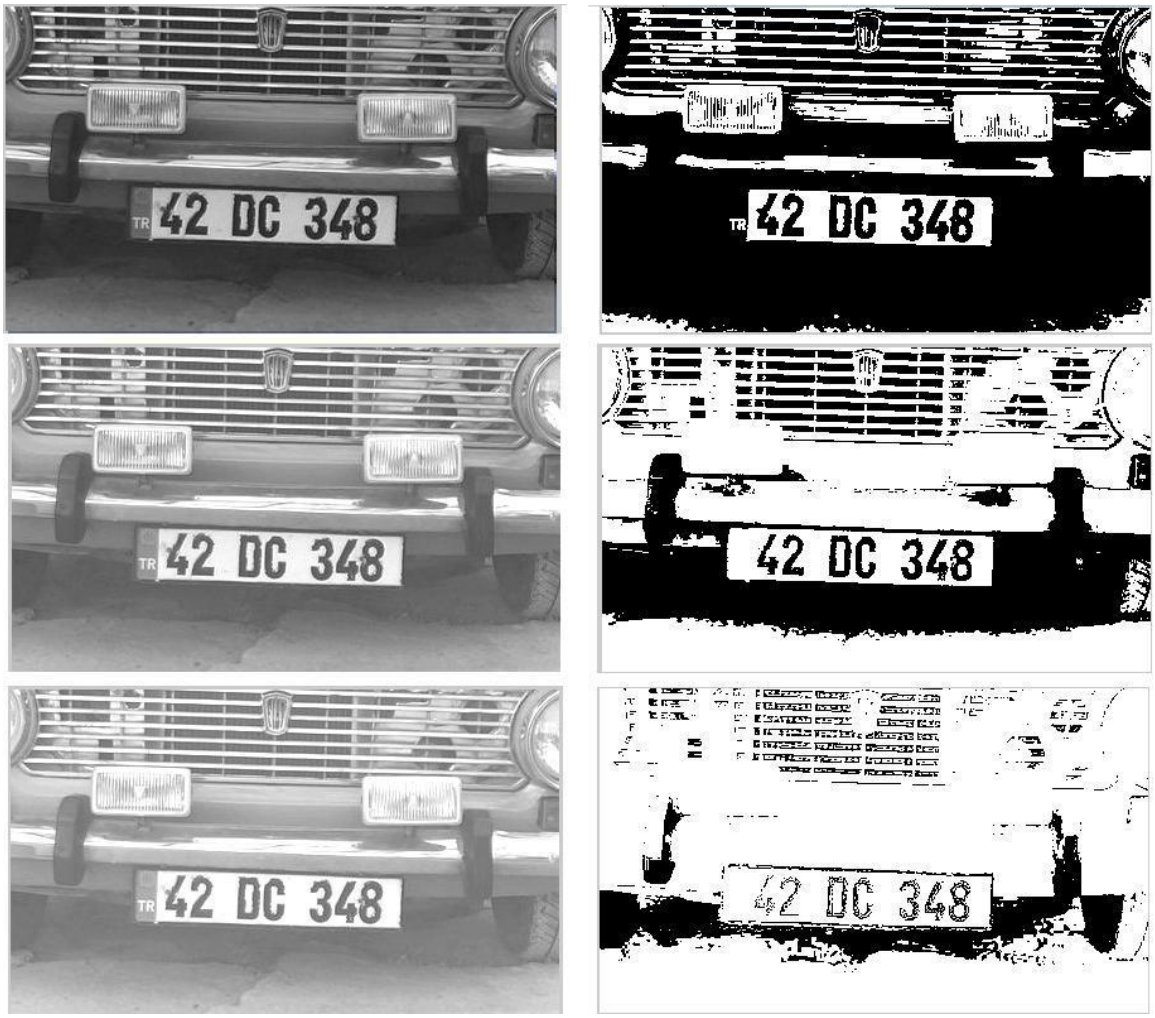


Şekil 3.25. Histogram eşitleme örneği ve diyagramı
a) Histogram eşitlemeden önce resim ve diyagram,
b) Histogram eşitledikten sonra resim ve diyagram

3.6.4. Gri seviyeli resmin siyah - beyaz resme dönüşümü

Resim işleme, görüntüyü oluşturan piksellerin durumları göz önünde bulundurularak gerçekleştirilen bir olaydır. Resim üzerindeki nesneler, piksellerin birbirleriyle olan ilişkileri ve değerleri hesaplanarak bulunabilir. Bu işlemler sırasında bir piksel değeri ne kadar karmaşık yapıda olursa yapılacak hesaplamalar o kadar çoğalır ve zorlaşır. Görüntünün hızlı ve etkili bir şekilde işlenmesi için, görüntüyü oluşturan piksel değerlerinin en sade hale getirilmesi gerekir. Bu yüzden bir piksel değerinin en sade ve yalın biçim olan siyah ya da beyaz hale getirilmesi gerekir. Siyah beyaz resim 0 ve 1 değerlerinden oluşan resimlere verilen format ismidir. Resim üzerinde cisim aramadan önce siyah beyaz formata dönüştürülmesi hız bakımından büyük katkı sağlar. Bir gri resim her pikseli için 0 ile 255 arasında değerler alır. Siyah-beyaz resimde 0 ile 1 arasında değerler aldığına göre, 0 ve 127 arasındaki değerleri 0, 127 ve 255 arasındaki değerleri 1 olarak değiştirerek, gri resimden siyah beyaz resim elde edilmiş olur. Fakat gri resimlerin parlaklık değeri, çekildiği ortama, atmosfer koşullarına ve çevresel koşullara göre farklılık göstereceği için bu sınır (eşik) değerinin değişken olarak belirlenmesi gerekir. Çünkü yüksek ışık bulunan bir ortamda çekilen gri bir resmin piksel değerleri daha çok beyaz renge (0 değerine) yakın olacaktır. Tam tersi bir durum olan, karanlık bir ortamda çekilen resmin piksel değerleri daha çok siyah renge (255 değerine) yakın olacaktır. Bu durumlarda, sınır (eşik) değeri olarak 127 sayının alınmasıyla siyah beyaz formata dönüştürülen görüntü üzerinde netlik ve kalite kaybolacaktır [29].

Eğer eşik değeri belirlenmeden 0-127 arasındaki değerler 0, 127-255 arasındaki değerler 1 olarak siyah-beyaz resme çevirme işlemi yapılırsa sonuç Resim 3.6’da görüldüğü gibi olur [25].



Resim 3.6. Sabit 127 olan eşik değeriyle gri seviyeli resmin - siyah beyaz resme dönüşümü [25]

Otsu'nun bölgesel eşikleme algoritması değişken eşik seviyesini bulmak için kullanılabilir. Bulunan eşik seviyesi [0,1] aralığında, parlaklık parametresidir. Otsu metodu gri seviye için eşikleme yaparken eşikleme değerini optimum olarak bulur. Bir kere histogram değeri hesaplandıktan sonra yöntem oldukça basit ve hızlıdır. Otsu metodunun gri seviyeden iki seviyeli imgeye dönüştürürken yaptığı varsayımlar [29]:

- Histogram ve imgenin bimodal (siyah ve beyaz) olduğu
- İmgenin sabit istatistik değerlere sahip ve değişmeyen parlaklık değerine sahip olduğu, şeklindedir.

Metot, gri seviye imgenin piksellerini iki tane sınıfa ayırır. Bu sınıflar arka plan ve nesnedir. Bunun için resmin önce histogramı çıkarılır.

Histogram çıkarıldıktan sonra otsu algoritmasıyla eşik değeri belirlenir. Eşik değeri bulunduktan sonra, bulunan eşik değerinin üstünde kalan piksel değerleri 1, eşik değerinin altında kalan piksel değerleri ise 0 olacak şekilde siyah beyaz resme dönüşüm işlemi yapılır [25].

Resim 3.7. Otsu metoduyla eşik değeri hesaplandıktan sonra gri seviyeli resmin siyah beyaz resme dönüşümü [25]

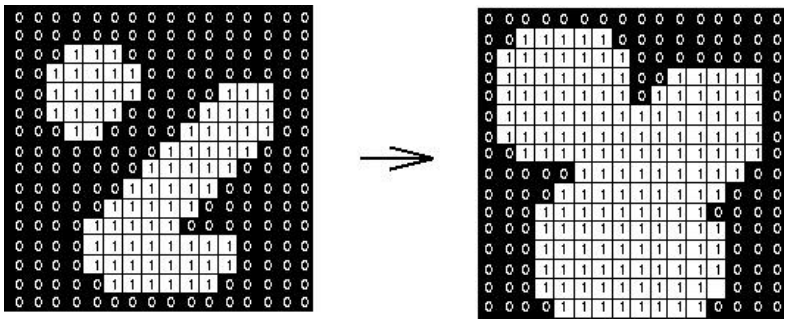
Resim 3.6. ve Resim 3.7.'de görülen siyah beyaz resimler arasındaki görüntü farkı çok rahat anlaşılmaktadır. Resimleri siyah beyaz resme çevirmeden önce eşik değerlerinin hesaplanmasının ne kadar önemli olduğu bu resimlerden anlaşılabilir. Kaliteli bir siyah beyaz resim üzerinden işlem yapmak çok daha sağlıklı, doğru ve hızlı sonuç alınmasını sağlar [25].

3.6.5. Morfolojik işlemler

Morfolojik imge işlemede temel olarak kullanılan iki işlem vardır: Genişletme (dilation) ve Aşındırma (erosion). Diğer morfolojik işlemler, bu temel iki işlem kullanılarak elde edilir [30].

Genişletmek (Yayma): İkili imgedeki nesneyi büyütmeye ya da kalınlaştırmaya yarayan morfolojik işlemdir. Sayısal bir resmi genişletmek demek resmi yapısal elemanla kesiştiği bölümler kadar büyütmek demektir. Kalınlaştırma işleminin nasıl yapılacağı yapı elemanı (structure element) belirler. Yapısal eleman resim üzerinde piksel piksel dolaştırılır. Eğer yapısal elemanın orijini resim üzerinde "0" değerli bir piksel ile karsılaşırsa herhangi bir değişiklik meydana gelmez. Eğer değeri "1" olan bir piksel ile karsılaşırsa yapısal elemanla yapısal elemanın altında kalan pikseller mantıksal "veya" işlemine tabi tutulurlar. Yani herhangi "1" değeriyle sonuç "1" e çevrilir. Genişletme ile resim üzerindeki nesneler şişer. Nesne içinde delikler var ise bunlar kapanma eğilimi gösterirler. Ayrık nesneler birbirine yaklaşır ya da bağlanır [30].

Şekil 3.26'da 3x3 yapısal elemanı ile sayısal resim üzerine genişletme işleminin uygulanması gösterilmiştir. 3x3 lük yapısal elemanın tüm değerleri "1" dir [30].

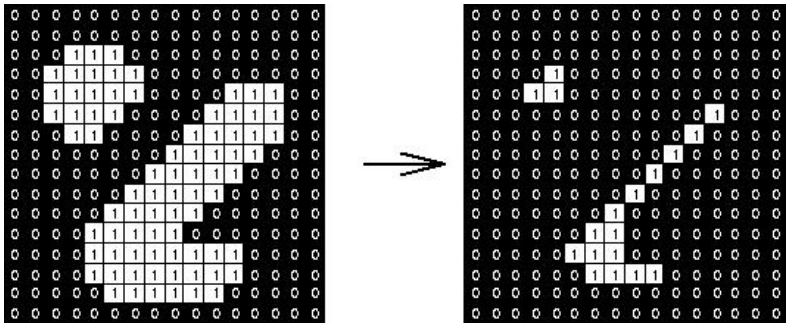


Şekil 3.26. Genişletme işlemi [30]

Aşındırma işlemi, ikili imgedeki nesneyi küçültmeye ya da inceltmeye yarayan morfolojik işlemdir. Aşındırma işlemi bir bakıma genişletmenin tersi gibi görülebilir. Burada yine aynı şekilde yapısal eleman resim üzerinde piksel piksel dolaştırılır fakat bu defa yapısal elemanın merkez pikseli "1" değeri ile karsılaşırsa yapısal eleman içerisindeki piksellerin durumuna bakılır.

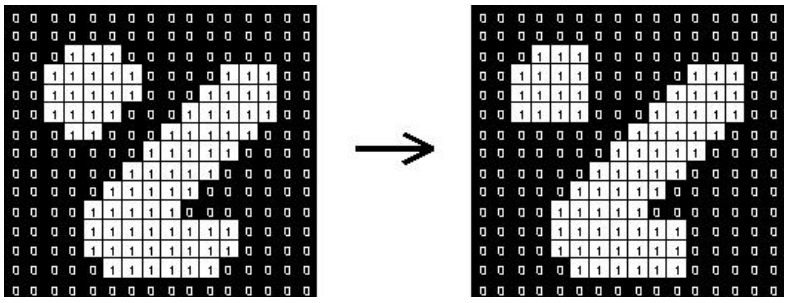
Eğer yapısal eleman içerisindeki "1" olan piksellerden herhangi biri altında resme ait "0" değeri varsa yapısal elemanın diğer "1" lerinin altındakilerle beraber bu piksel "0"'a dönüştürülür. Aşındırma işlemi ile sayısal resim aşındırılmış olur. Yani resim içerisindeki nesneler ufalır, delik varsa genişler, bağlı nesneler ayrılma eğilimi gösterir [30].

Şekil 3.27'de 3x3 yapısal elemanı ile sayısal resim üzerine aşındırma uygulanması gösterilmiştir. 3x3 lük yapısal elemanın tüm değerleri "1" dir [30].



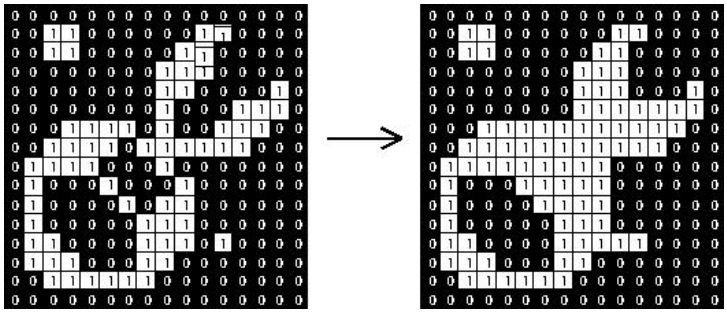
Şekil 3.27. Aşındırma işlemi [30]

Açma işlemi, sayısal bir resme önce aşındırma daha sonra genişletme uygulanırsa resme açma işlemi uygulanmış olur. Açma işlemine tabi tutulmuş bir görüntü ve değişimi Şekil 3.28'de gösterilmiştir. Burada yine 3x3 lük yapısal eleman kullanılmıştır [30].



Şekil 3.28. Açma işlemi [30]

Kapatma işlemi, sayısal resme önce genişletme daha sonra aşındırma uygulanırsa Kapatma işlemi uygulanmış olur. Şekil 3.29'da kapatma işlemi uygulanmış bir görüntünün öncesi ve sonrası görülmektedir [30].



Şekil 3.29. Kapatma işlemi [30]

Eğer morfolojik işlemin sonucunda resimdeki nesnelerin keskin hatları silinip yerlerine kavisli veya daha yumuşak hatlar getirilmek isteniyorsa dairesel yapısal eleman kullanılmalıdır. Örneğin erozyon işleminde resim içerisindeki nesnelerin en ve boyları aynı oranda azaltılmak (erozyona uğratılmak) isteniyorsa yapısal eleman kare seçilmelidir [30]

3.6.6. Karakter ayrıştırma

Plaka bölgesi belirlendikten sonra karakterler tanımlanmak üzere tek tek ayrıştırılmalıdır. Bu işlem için kullanılabilecek en etkin yöntemler “karakterler arası boşluk takibi” ve “ dikey izdüşüm metodu” dur.

Karakterler arası boşluk takibi

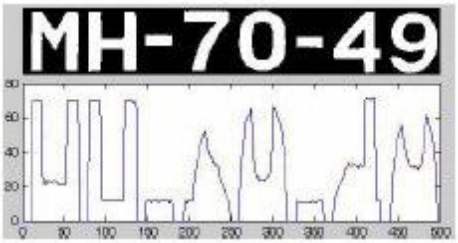
Bu metot plaka üzerindeki karakterler arasındaki boşlukları sınır noktaları olarak kullanmaktadır. Yapılandırılan işlem bir tarayıcı çizgi yardımıyla, siyah/beyaz formundaki plaka resmi üzerinde soldan başlayarak “siyah” renkli piksel taşımayan tüm kolonların işaretlenmesi ile yapılmaktadır [31].

Şekil 3.30. Plaka üzerindeki harfler arasında boşluk takibi [31]

Bu yöntem çok temiz elde edilmiş siyah/beyaz plaka görüntüleri üzerinde basit ve hızlı bir çözüm üretmektedir. Yani bu basit metodun başarılı olabilmesi için önceki adımlardan iyi kalitede bir plaka görüntüsü bu aşamaya aktarılmış olmalıdır [31].

Dikey izdüşüm yöntemi

Plakadaki karakterlerin ayrıştırılması için diğer bir yöntem de plakadaki karakterlerin dikey olarak izdüşümlerinin çıkarılması ve bunların takibidir. Plakanın yerleştirilmesinden sonra, karakterleri ayrıştırmak için dikey bir izdüşüm yapılmıştır. Resim 3.8'de görüldüğü gibi, izdüşüm konturu bu karakterler arasındaki yüksek pozitif değeri ve aralıklar arasındaki çok düşük pozitif değerleri göstermektedir. Düşük değer alanlar arasındaki mesafe karakterin genişliği olarak kabul edilebilir [31].



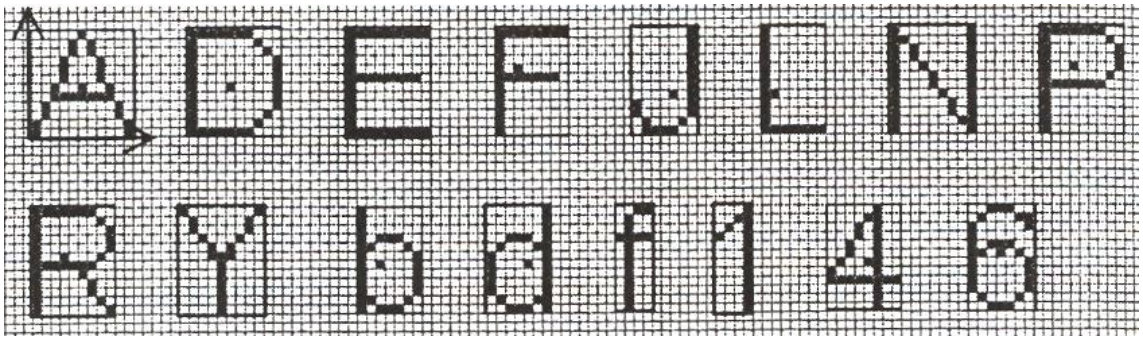
Resim 3.8. İzdüşüm yöntemi ile karakterlerin ayrıştırılması [31]

3.6.7. Karakter tanıma yöntemleri

Plakadaki karakterler ayrıştırıldıktan sonra tanımlama işlemi başlar. Ağırlık merkezli tanımlama, şablon eşleştirme gibi karakter tanıma teknikleri kullanılır.

Ağırlık merkezine dayalı tanıma

İsminden de anlaşıldığı gibi bu yöntem karakterlerin ağırlık merkezinin hesaplanmasına dayalıdır. Resim 3.9'da bazı karakterlerin ağırlık merkezlerine (G(x,y)) işaret edilmiştir. Piksel bazında incelendiğinde ağırlık merkezlerinin aynı noktada olduğu söylenebilir. Oysaki oransal temelde bakıldığında bu değerler farklı olacaktır. Ağırlık Merkezine Dayalı sınıflandırma sabit karakter boyutlarında (bazı çakışmalar göz ardı edilirse) iyi sonuçlar vermektedir [26].



Resim 3.9. Bazı karakterlerin ağırlık merkezleri [26]

Matematiksel olarak, çerçevelenmiş bir karakterin (x,y) koordinat sisteminde G(x,y) ağırlık merkezinin bileşenleri aşağıdaki bağıntılarla bulunur.

$$G_x \frac{\sum_{J=1}^{en} \sum_{i=1}^{boy} F_x\ i,j}{siyah\ piksel\ sayısı} \quad ve \quad G_y \frac{\sum_{i=1}^{boy} \sum_{j=1}^{en} F_y\ i,j}{siyah\ piksel\ sayısı} \tag{3.4}$$

Farklı büyüklükteki karakterleri karşılaştırabilmek için G(x,y), normalize edilmelidir. Normalize edilmiş G'(x,y) ağırlık merkezinin bileşenleri;

$$G_x' = \frac{G_x}{en} \quad ve \quad G_y' = \frac{G_y}{boy} \tag{3.5}$$

bağıntılarıyla bulunur.

Ağırlık merkezi, her ne kadar ayırt edici özellik taşısa da karakterlerin büyüklüğüne çok duyarlıdır. Karakterlerde ölçekleme işlemleri piksel bazında yapıldığından ağırlık merkezinin yeri de değişmektedir. Hatta aynı yazı türünün farklı büyüklüklerdeki karakterlerinin ağırlık merkezleri de farklı olabilir. Bu nedenle çeşitli özellik vektörlerine gerek duyulmaktadır [26].

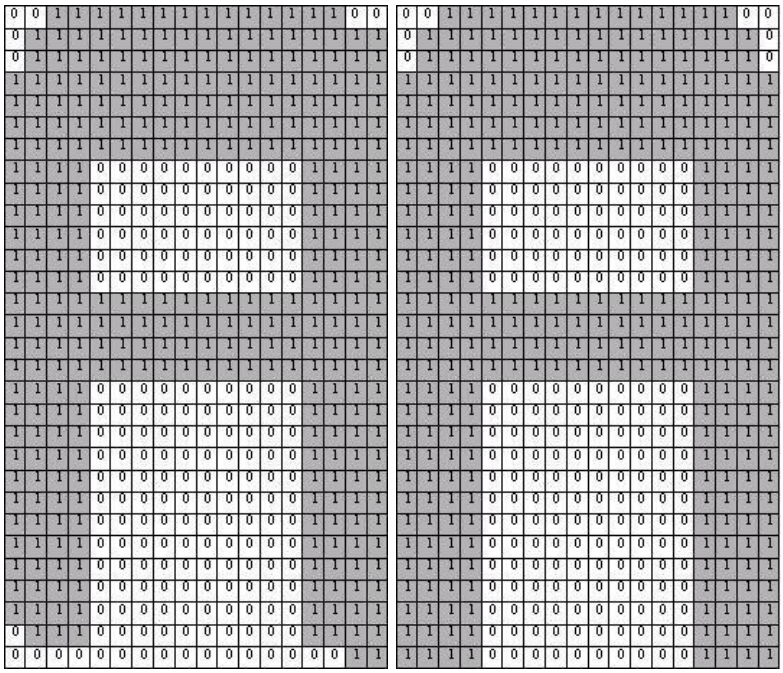
Şablon eşleştirme yöntemi

Şablon eşleştirme (template matching) yönteminde karakterlerine ayrılmış plakadaki her karakter resmi daha önceden veri tabanına kayıtlı karakterlerle kıyaslanır ve birbirlerine benzerliği bulunmaya çalışılır [31].

Şablon esleştirme yönteminden yüksek basarı elde etmenin yolu aynı karakterlerden şablon kütüphanesinde (veritabanında) fazla sayıda bulunmasıdır. Bu şekilde bir karakter daha fazla şablon ile karşılaştırılacak ve daha yakınsak bir benzerlik elde etmek mümkün olacaktır. Ancak veritabanının çok fazla sayıda örnek karakter (şablon) içermesi karşılaştırma süresinin uzamasına dolayısıyla sistemin yavaşlamasına neden olacaktır [31].

Şablon eşleştirme yönteminde plaka resmi üzerinde ayrıştırılan karakter resmi veritabanındaki şablonlarla piksel piksel karşılaştırılır ve en iyi sonucu üreten yani en çok benzeyen şablon resmin karşılığı olan karakter olarak kabul edilir [31].

Şablon esleştirme yöntemi için gerekli olan en önemli önkoşul şablonların boyutlarının birbirine eşit olmasıdır. Plaka üzerinden kopartılmış karakter resminin de şablonların boyutlarına eşitlendikten sonra karşılaştırma işleminin yapılması gerekir. Aksi takdirde sistemin başarılı sonuçlar üretmesi mümkün değildir. Sistemin başarılı oranının yüksek olmasını sağlayan diğer önemli bir etmen şablon boyutlarının büyük tutulmasıdır [31].



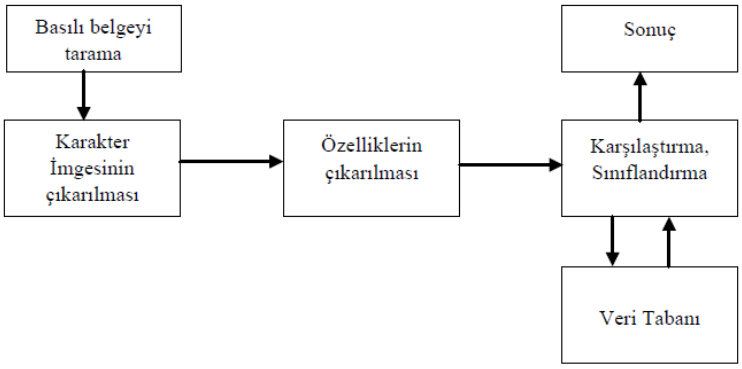
(a) (b)

Şekil 3.31. Şablon esleştirme yöntemi [31]
a) Plaka üzerinden ayrıştırılan karakter resmi
b)Veritabanında karşılaştırma sonucu bulunan karakter

3.6.8. OCR (optik karakter tanıma) metodu

İmge tanıma teknolojilerinin alanlarından birisi olan OCR işlemi, basılı metinlerin bir tarayıcı vasıtası ile bilgisayar ortamına aktarıldıktan sonra ya da bir kameradan alınan dijital görüntüdeki yazıların metinlere dönüştürülmesi işlemidir. Bu işlem basılı metinler üzerinde çalışan birimler için vazgeçilmez bir teknolojidir. Doküman işleme işlemleri ile çalışırken maliyetleri düşürmek ve en yüksek düzeyde verim elde etmek için OCR teknolojilerinden faydalanmak kaçınılmaz olmuştur. Son yıllarda trafik denetimlerinde ve otoparklarda da OCR tekniğinden faydalanılarak plaka tanımlama işlemleri yapılmaktadır [32].

OCR sistemleri, genel olarak basılı belgeyi tarar, taranan belgede bulunan metindeki karakter imgelerini ayırır ve önceden belli olan karakter şekilleri ile karşılaştırma yaparak bu imgenin hangi karaktere ait olduğunu tespit eder. Sistem, önceden belli olan karakter imgelerini kendi veritabanında tutar. Optik Karakter tanıma sistemlerinin genel yapısı Şekil 3.32'de verilmiştir [32].



Şekil 3.32. OCR sisteminin genel yapısı [32]

Karakter imgesi modülünde, metnin taranması ve karakterlerin imgelerinin ayırtılarak sistem girişine verilmesi işlemleri yapılır. Sonraki adımda karakter imgesinin şekilsel özellikleri çıkartılır ve veri tabanında olan karakter imgelerinin şekilsel özellikleri ile karşılaştırılarak hangi karakter olduğuna karar verilir. Her karakter, farklı ölçülerde ve şekillerde gösterilebilir. Karakterlerin karşılaştırılması, önceden sisteme tanımlı olan ve veri tabanında bulunan karakter imgelerine göre sınıflandırılması ile gerçekleşir. Bu işlemi yapan modüle sınıflandırıcı denir. Sistemin çıkışı, tanınan karakterin kodudur [32].

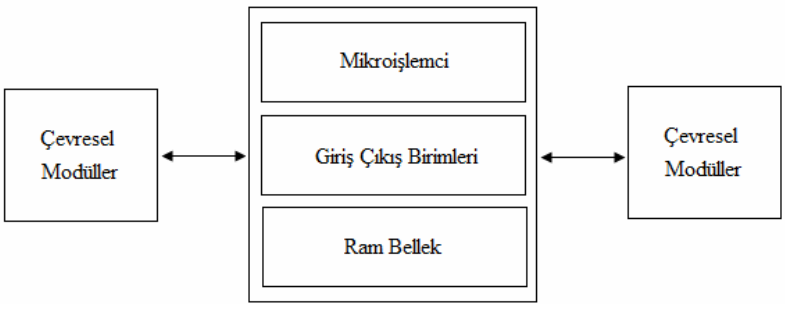
4. PIC MİKRODENETLEYİCİLERİ

Bu bölümde mikrodenetleyiciler hakkında genel bilgi verilmiş ve uygulamanın elektronik kontrol devresinde kullanılan PIC 16F877A mikrodenetleyicisinin özellikleri incelenmiştir.

4.1. Mikrodetleyicilere Genel Bakış

Sistemlerin kontrol edilmesi yakın zamana kadar analog sistemler ile gerçekleştirilirken günümüzde analog sistemlerin yerini hem sayısal hem de analog sistemler almıştır. Sayısal sistemlerin hızla gelişmesi, yapılması istenen işlemlerin daha küçük sayısal devreler yardımıyla geliştirilme fikrinin oluşmasını sağlamıştır. Bu durumdan hareket eden üreticiler küçük sistemlerle daha fazla işlem gerçekleştirebilmek için sistemleri en küçük boyutlarda üretmeye çalışmaktadırlar. Bu nedenlerden dolayı mikrodenetleyicilerin kullanımı yaygınlaşmıştır. Mikrodenetleyiciler, 3 ayrı blok halinde bulunan tümleşik (entegre) elemanlardır [33].

Mikroişlemcilerin kullanımı birçok açısında dönüm noktası olmuş, ilerleyen teknolojiye bağlı olarak mikroişlemcilerin ve çevresinde bulunan donanımlarının hacimsel ebatları küçülme eğilimine gitmiştir. İste tam da bu noktada tümleşik devre talebine cevap verebilmek için bu sistemleri tek bir entegre içinde birleştirme fikri oluşmuştur. Bu tümleşik devrelere mikrodenetleyici adı verilmiştir. Böylelikle giriş-çıkış birimi, mikroişlemci ve hafıza alanını bir arada bulunduran bu tümleşikdevrelerin seri imalatına geçilmiştir. Şekil 4.1'de mikrodenetleyicinin genel yapısı görülebilir [33].



Şekil 4.1. Mikrodenetleyicinin genel yapısı

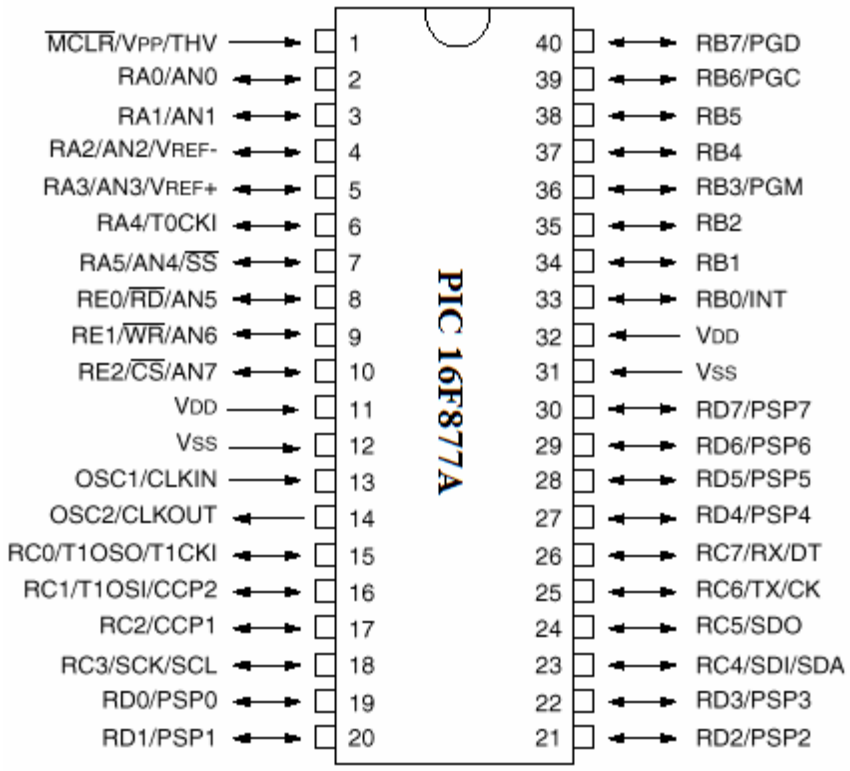
Şekil 4.1'de görüldüğü gibi mikrodetleyicilerin iç yapısını oluşturan mikroişlemci, giriş/çıkış birimi, rastgele erişimli bellek (Random Access Memory-RAM) ile bu mikrodenetleyici ile karşılıklı haberleşme sağlayacak çevre birimleri yer almaktadır. Mikrodetleyicilerin temel çalışma mantığı ise şöyledir: algılayıcı veya buton gibi çevresel modüllerden gelen dijital bilgiyi giriş birimlerinden alarak bellek aracılığıyla mikroişlemciye iletir. Mikroişlemciye gelen bilgi programın içeriğine göre işlenip, çıkış birimleri aracılığıyla motor, röle gibi çevresel birimlere gönderilir [33].

4.2. Otomatik Otopark Sisteminde Kullanılan Mikrodenetleyicinin Seçimi

Mikrodenetleyici üreten çeşitli elektronik firmaları bulunmaktadır. Bu firmalar kullanıcıların sıklıkla tercih ettikleri istekleri yerine getirmek üzere değişik özelliklere sahip mikrodenetleyiciler üretirler. Kullanıcılar da tasarımını yapmayı düşündükleri sistemlere yönelik olarak belirlemiş oldukları özellikleri karşılayabilecek mikrodenetleyicileri tercih ederler [33].

Otomatik otopark sisteminin tasarımında Microchip firmasının üretmiş olduğu PIC 16F877A mikrodenetleyicisi tercih edilmiştir. Peripheral Interface Controller (Çevresel Arayüz Kontrolcüsü) kelimelerinin baş harflerinden oluşan PIC, önceleri Programmable Interface Controller olarak sonraki yıllarda Programmable Intelligent Computer olarak da adlandırılmıştır. Bu mikrodenetleyicinin tercih edilme sebepleri söyle sıralanabilir [33]:

a) Giriş çıkış sayısının tasarım için yeterli olması: Otomatik otopark sisteminin tasarımında dışarıdan gelen verilerin değerlendirilip daha sonra çevresel birimlere aktarılabilmesi için giriş ve çıkış birimlerine ihtiyaç bulunmaktadır. Projeye bakıldığı zaman 19 giriş portuna, 7 çıkış portuna ihtiyaç duyulduğu görülmektedir. Şekil 4.2'de PIC 16F877A mikrodenetleyicisinin bağlantı uçları görülmektedir.



Şekil 4.2. PIC 16F877A mikrodenetleyicisinin bağlantı uçları

Şekil 4.2'den anlaşılacağı üzere PIC 16F877A' nın PortA' ya ait 6 adet, PortB' ye ait 8 adet, PortC' ye ait 8 adet, PortD' ye ait 8 adet, PortE' ait 3 adet olmak üzere toplam 33 adet giriş ve çıkış ucu bulunmaktadır. Otomatik otopark sisteminin tasarımında kullanılması düşünülen giriş ve çıkış sayısı ise 26'dır. Bu değerlere bakıldığı zaman otomatik otopark sisteminin elektronik kontrol devresinin tasarımında PIC 16F877A'nın kullanılmasının doğru olacağı düşünülmüştür.

b) Kullanım alanının genişliği: Kullanıcıların tasarımlarını gerçekleştirirken mikrodenetleyicileri tercih etme nedenlerinden biri de sektördeki kullanım alanın genişliğidir. Bir mikrodenetleyici ne kadar çok kullanıcı tarafından tercih edilirse üreticiler de o mikrodenetleyiciye ait özellikleri daha kısa zamanda geliştirirler. Piyasada Microchip firmasının üretmiş olduğu mikrodenetleyiciler oldukça sık tercih edilmektedir. 16F877A mikrodenetleyicisinin tercih edilme sebepleri arasında teknik doküman desteğinin yeterli olması, elektronik devre elamanlarının bu mikrodenetleyici ile uyumlu çalışması, temininin kolay olması ve bu mikrodenetleyici grubunun birçok yazılımı desteklemesi gösterilebilir.

c) Program belleğinin ideal boyutlarda olması: PIC 16F877A' nın 8KB'lık program beleği vardır. Her bellek biriminde 14 bit uzunluğundaki program komutları kaydedilir. Ayrıca EEPROM veri belleği 256 byte, kullanıcı RAM 368x8 byte boyutundadır.

d) Programın yazılacağı yazılımın desteklemesi: Elektronik devrelerde mikrodenetleyici kullanılacaksa bir programlama dili tercih edilmelidir. Mikrodenetleyici programı, seçilen programlama diline bağlı olarak bir derleyici program aracılığıyla yapılır. Bu derleyicinin desteklediği mikrodenetleyici marka ve modelleri programın yazılması, derlenmesi, benzetimin gerçekleştirilmesi ve ardından da mikrodenetleyiciye yüklenmesi açısında önem teşkil etmektedir.

Otomatik otopark sisteminin tasarımında kullanılacak mikrodenetleyicinin programı CCS C adlı program aracılığıyla yapılmıştır. Bu programın listesinde PIC 16F877A'nın da bulunduğu yazılımın kolay temin edilebilmesi nedeniyle PIC 16F877A tercih edilmiştir.

e) Programın derlenmesinden sonra benzetimi gerçekleştirmek için kullanılacak programın desteklemesi: Mikrodenetleyici programının yazılması ve derlenmesinden sonra programın derlenmiş halini mikrodenetleyiciye yüklemeden önce programın çalışmasını sanal ortamda kontrol etmekte fayda vardır. Bu kontrol işlemi simulasyon programları aracılığı ile yapılır. Bu programlar aracılığıyla sistemin gerçek zamanda karşılaşacağı sorunların daha önceden test edilerek belirlenir. Sorunlar düzeltildikten sonra programın mikrodenetleyiciye yüklenmesi sağlanabilir. Bu programlar çeşitli mikrodenetleyici marka ve modelinin simulasyonunu yapabilmektedir.

f) Gerçek zamanda programı mikrodenetleyiciye yükleyebilme kolaylığı: Programın yazılmasının ardından derlenmiş halininin mikrodenetleyiciye aktarılması gerekmektedir. Bu yükleme işlemi mikrodenetleyiciyi destekleyen kartlar ve arayüz programları ile yapılmaktadır.

PIC 16F877A'yı destekleyen arayüz ve yükleme kartlarının piyasada ve internet ortamında kolaylıkla bulunabilmesi otomatik otopark sisteminin programlanmasında PIC 16F877A'nın tercih edilmesini sağlamıştır.

4.3. PIC 16F877A Mikrodenetleyicisi

PIC 16F877A, geçmişteki en popüler PIC işlemcisi olan PIC 16F84'den sonra kullanıcılara yeni ve gelişmiş olanaklar sunmasıyla hemen dikkat çekmektedir. PIC 16F877A'nın program belleği FLASH bellek olup tıpkı PIC 16F84'de olduğu gibi elektriksel olarak defalarca silinip yazılabilmektedir.

Özellikle PIC 16C6X ve PIC 16C7X ailesinin tüm özelliklerini barındırması, PIC 16F877A'yı kod geliştirmede de iyi bir çözüm olarak sunmaktadır. Yapılandırma (configuration) bitlerini göz önüne almak şartıyla C6X veya C7X ailesinden herhangi bir işlemci için geliştirilen program hemen hiçbir değişikliğe gerek kalmadan 16F877A'ya yüklenebilir ve çalışmalarda denenebilir. Bunun yanı sıra PIC 16F877A, PIC 16C74 ve PIC 16C77 işlemcileriyle pinleri bire bir uyumludur [33].

4.4. PIC 16F877A Mikrodenetleyicisinin Portları

PIC 16F877 mikrodenetleyicisinde A, B, C, D, E olmak üzere 33 giriş/çıkış portu vardır. Bu portlar isteğe göre giriş ya da çıkış olarak ayarlanabilir. Hangi portun hangi bitlerinin giriş hangi bitlerinin çıkış olarak kullanılacağı programın başında ayarlanmalıdır. Giriş ve çıkış olarak belirtilen bu portların birden çok görevi olabilir. Hangi amaçla kullanılmak isteniyorsa gerekli ayarlamalar yapılarak program ona göre hazırlanmalıdır. Portların sahip olduğu pinler birden fazla görevi yerine getirmek üzere tasarlanmış olabilir. Bu nedenle programlama aşamasında hangi pinin hangi görevde kullanılacağı bilinmeli ve program ona göre hazırlanmalıdır [33].

PORTA 6 bitlik hem giriş hem de çıkış özelliği olan bir porttur. Hangi bitin giriş hangi bitin çıkış olacağı TRISA kaydedicisi yardımıyla belirlenir. TRISA kaydedicisinde "1" olarak belirlenen bitlerin karşılığı giriş, "0" olarak belirlenen

bitlerin karşılığı çıkış olmaktadır. PORTA tamponlanmıştır yani yeni bir veri gönderilene kadar eski veri PORTA kaydedicisinde tutulur. Mikrodenetleyiciye gerilim uygulayıp çalışmaya başlattığımızda yani POR (Power-On Reset) durumunda PORTA default olarak analog giriştir. Okunmak istendiğinde 0 bilgisini verir. Sayısal giriş olarak kullanılmak istendiğinde ADCON1 kaydedicisinde gerekli ayarın yapılması gerekir.

PORTB hem giriş hem çıkış özelliği olan 8 bitlik bir portttur. PortB'nin en çok göze çarpan özelliği RB0 kesmesi ve RB4 - RB7 arasındaki girişlerde bilgi değişikliğinde oluşan kesme durumudur. RB0 kesme girişi olarak kurulduğunda isteğe başlı olarak yükselen kenarda veya düşen kenarda bir kesme üretebilmektedir. RB4 – RB7 arasında pinlerde bulunan bilgilerden biri değiştiğinde de kesme oluşabilmektedir. PORTB pinleri programlama ve hata ayıklama dışındaki amaçlarda kullanıldıkları sürece PORTA' da olduğu gibi TTL (Transistor-Transistör Mantık) gerilim seviyelerinde çalışır.

PORTC, PIC16F877A'nın en çok özelliği barındıran portudur. Tüm girişler schmitter-trigger tampona sahiptir. Bunun nedeni pinlerin değişik seri haberleşme fonksiyonlarına sahip olmasıdır.

PORTD ve PORTE genelde birlikte kullanılır. Mikro bilgisayar veri yollarıyla 8 bitlik paralel iletişim için kullanılırlar. PORTD 8 bit'lik veri ve adres yolunu bilgilerini taşırken, PORTE kontrol bilgilerini taşır. Tüm girişler paralel iletişim sırasında TTL seviyelerinde, çıkış olarak kullanıldıklarında ise Schmitt- Trigger seviyelerde çalışır. PORTE aynı zamanda PORTA gibi analog giriş olarak da belirlenebilmektedir.

4.5. 16F87X Mikrodenetleyici Ailesinin Genel Özellikleri

- CPU azaltılmış komut seti
- RISC temeline dayanır.
- Öğrenilecek 35 komut vardır ve her biri 14 bit uzunluktadır.
- Dallanma komutları iki çevrim (cycle) sürede, diğerleri ise bir çevrimli sürede uygulanır.

- İşlem hızı 16F877'de DC20 MHz'dir. (16F877A'da bir komut 200ns hızında çalışır.)
- Veri yolu (databus) 8 bittir.
- 32 adet SFR (Special Function Register) olarak adlandırılan özel işlem kaydedicisi vardır ve bunlar statik RAM üzerindedir.
- 8 KB flash belleği 1 milyon kez programlanabilir.
- 368 Byte veri belleği (RAM),
- 256 Byte EEPROM veri belleği vardır.
- Pin çıkışları PIC 16C73B/74B/76 ve 77 ile uyumludur.
- 14 kaynaktan kesme yapabilir.
- Yığın derinliği 8'dir.
- Doğrudan, dolaylı ve göreceli adresleme yapabilir.
- Poweron Reset (Enerji verildiğinde sistemi resetleme özelliği)
- Powerup Timer (Powerup zamanlayıcı)
- Osc. Startup Timer (Osilatör başlatma zamanlayıcısı)
- Watchdog Timer (Özel tip zamanlayıcı), devre içi RC osilatör
- Programla kod güvenliğinin sağlanabilmesi özelliği
- Devre içi Debugger (Hata ayıklamakta kullanılabilecek modül)
- Düşük gerilimli programlama
- Flash ROM program belleği (EEPROM özellikli program belleği)
- Enerji tasarrufu sağlayan, sleep modu
- Seçimli osilatör özellikleri
- Düşük güçle, yüksek hızla erişilebilen, CMOS Flash EEPROM teknoloji
- Tümüyle statik tasarım
- 2 pinle programlanabilme özelliği
- Yalnız 5V girişle, devre içi seri programlanabilme özelliği
- İşlemcinin program belleğine, okuma/yazma özelliği ile erişimi
- 2.0 V – 5.0 V arasında değişen geniş işletim aralığı
- 25 mA'lik kaynak akımı
- Devre içi, iki pin ile hata ayıklama özelliği
- Geniş sıcaklık aralığında çalışabilme özelliği
- Düşük güçle çalışabilme özelliği.

4.6. Çevresel Özellikler

- TMR0: 8 bitlik zamanlayıcı, 8 bit ön bölücülü
- TMR1: Ön bölücülü, 16 bit zamanlayıcı, uyuma modundayken dış kristal zamanlayıcıdan kontrolü arttırılabilir.
- TMR2: 8 bitlik zamanlayıcı, hem ön bölücü hem de son bölücü sabiti
- İki Capture / Compare / PWM modülü
- 10 bit çok kanallı A/D çevirici
- Senkron seri port (SSP), SPI (Master mod) ve I^2C (Master Slave) ile birlikte
- Paralel Slave Port, 8 bit genişlikte ve dış RD, WR, CS kontrolleri
- USART/SCI, 9 bit adres yakalamalı
- BOR Reset (Brown Out Reset) özelliği.

4.7. PIC16F877'nin Besleme Uçları ve Beslenmesi

PIC16F877'nin besleme gerilimi 11, 12 ve 31, 32 numaralı pinlerden uygulanmaktadır. 11 ve 32 numaralı Vdd uçları +5V'a ve 12, 31 numaralı Vss uçları toprağa bağlanır. Besleme uçlarının ikişerli olarak kullanımı zorunlu değildir. Örneğin besleme için 12 ve 32 nolu pinler kullanılıp 11 ve 31 nolu pinler boş bırakılabilir. PIC'e ilk defa enerji verildiği anda meydana gelebilecek gerilim dalgalanmaları neden olabileceği arızaları önlemek amacıyla 100nF'lık dekuplaj kondansatörünün + ve – besleme uçları arasına bağlanmasında fayda vardır. PIC'ler 2 ila 6 Volt arasında çalışabilmektedirler. +5V'luk bir gerilim ise ideal bir değer olmaktadır.

4.8. PIC16F877'nin Reset Uçları

Kullanıcının programın işleyişini isteyerek kesip başlangıca döndürebilmesi için 16F877A'nın 1 numaralı ucu MCLR (master clear) olarak tasarlanmıştır. MCLR ucundaki gerilim şase (GND) seviyesine çekildiğinde programın çalışması başlangıç adresine döner. Programın ilk başlangıç adresinden itibaren tekrar çalışabilmesi için, aynı uca +5 Volt gerilim uygulanmalıdır.

4.9. PIC16F877'nin Clock Uçları ve Osilatör Tipleri

PIC16CXX mikrodenetleyicilerinde 4 çeşit osilatör kullanılabilmektedir. Kullanıcı bu 4 osilatör çeşidinden birini seçerek iki yapılandırma bitini (FOSC1 ve FOSC2) programlayabilir. PIC16F877A'da osilatör uçları 13 ve 14 nolu pinlerdir. Hazırlanacak olan PIC programlarında kullanılan osilatör tipi PIC programının çalışma hızını ve hassasiyetini etkileyeceğinden kullanıcı, amacına uygun bir osilatör devresi kullanılmalıdır. Osilatör tipinin seçiminde dikkat edilecek bir başka husus ise, seçilecek olan osilatörün kullanılan PIC'in özelliğine uygun olmasıdır. Örnek verilecek olursa; en fazla 10 MHz çalışma frekansına sahip bir PIC16F877 için 20 MHz'lik bir osilatör kullanmak uygun değildir. Fakat daha düşük bir frekans değeri ile çalışan bir osilatör devresi tercih edilebilir [33].

5. OTOMATİK OTOPARK SİSTEMİ TASARIMI ve UYGULAMASI

Bu bölümde park etmek için gelen aracı algılayıp plaka tanımlamasını yapan ve en uygun bölmeye taşıyarak park eden “Otomatik Otopark Sistemi”nin tasarım ve uygulama aşamaları tek tek incelenmiştir. Uygulama aşamaları incelenirken park bölmelerinin tasarımı, taşıyıcı mekanizmanın tasarımı, elektronik kontrol ünitesinin tasarımı ve bilgisayar programının sistemle nasıl uyumlu çalıştığı anlatılmıştır. Kullanılan tüm parçaların resimleri çekilmiş ve teknik çizimlerle desteklenerek yapılacak olan sonraki çalışmalara referans olması amaçlanmıştır.

5.1. Tasarımda ve Uygulamada Kullanılan Malzemelere Genel Bakış

Sistemin tasarımına geçmeden önce kullanılacak malzemeler ve özellikleri belirlenmiştir. Otomatik otopark sistemi dört ana bölümden oluşmaktadır. Bu bölümler; araçların park edileceği bölmeler, taşıyıcı elektromekanik sistem, elektronik kontrol birimi ve bilgisayar yazılımıdır.

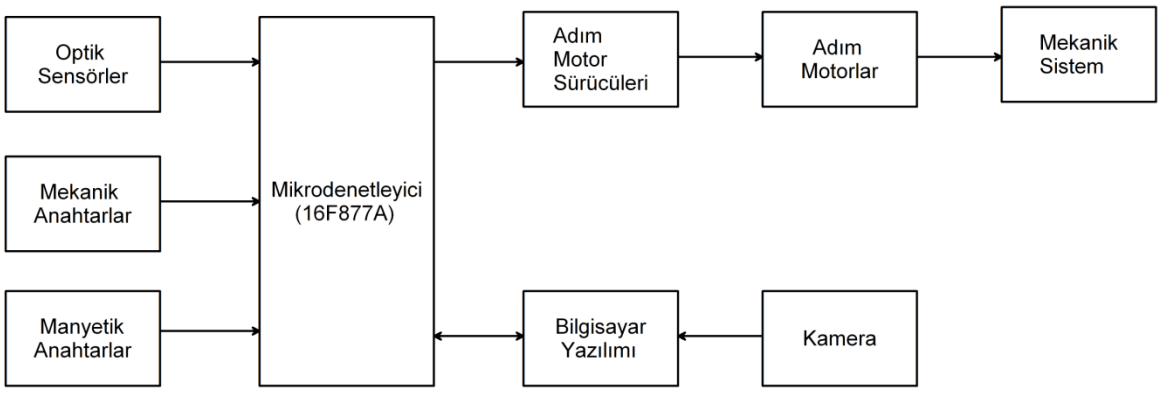
Araçların park edileceği bölmelerin tasarımında reklamcılık sektöründe de yaygın bir şekilde kullanılan dekota (plastech PVC foam yada foreks) adı verilen işlemesi kolay, hafif, esnek ve dayanıklı malzeme kullanılmıştır. Dekota maket bıçağı ile kolayca kesilebilen köpük PVC levhadır. 2mm ila 20mm arasında kalınlıklarda levha şeklinde üretilir. Taşıyıcı sistem ve park bölmeleri 5mm kalınlığında alüminyum kompozit malzeme üzerine yerleştirilmiştir.

Taşıyıcı elektromekanik sistemin tasarımında 1,5mm kalınlığında demir sac kullanılmıştır. Bu malzemenin seçilmesinin nedeni dayanıklı ve kaynak yapmaya uygun olmasıdır. Aslında taşıyıcı sistem 2 prizmatik 1 döner eksenden oluşan (R2P) bir silindirik koordinatlı robottur. Dikey hareket için bir adım motor ve bu motor ile kontrol edilen kayış-kasnak sistemi kullanılmıştır. Yatay hareket için ise yine bir adım motor ve bu motorun miline doğrudan bağlı bir palet kullanılmıştır. Yatay silindirik harekette sürtünmeyi azaltmak ve yükün tamamen motor miline binmesini önlemek için eksenel-sabit bilyalı rulman kullanılmıştır. İleri geri hareket için ise yine bir adım motor ve bu motorun miline bağlı pinyon dişli ile hareket eden kremayer dişli sistemi kullanılmıştır.

Adım motor seçilmesinin başlıca nedenleri; yüksek doğruluk pozisyonuna sahip olmaları, yüksek tork üretebilmeleri, mikrodenetleyici gibi sayısal kontrol birimleri ile kolaylıkla kontrol edilebilmeleri, ucuz sürücü devrelerinin piyasada kolaylıkla bulunabilmesi, uzun ömürlü olmaları ve bakım gerektirmemeleridir.

Elektronik kontrol birimi sistemin beyni olup tüm mekanizmayı kontrol etmektedir. Tek katlı bakır plaket üzerine yapılmıştır. Kontrol işlemi için Microchip firmasının 16F877A mikrodenetleyicisi kullanılmıştır. Bahsedilen mikrodenetleyici çok sayıda giriş/çıkış portuna sahip olması ve gerekli olan fonksiyonları yerine getirebilme özelliklerine sahip olması nedeni ile tercih edilmiştir. Ayrıca bilgisayar yazılımı ile iletişimi sağlamak için seri iletişim protokolü kullanılmış ve RS-232 portu ile arayüz oluşturması için MAX232 entegresinden faydalanılmıştır. Araçların park edileceği bölmelerin boş yada dolu olma durumunu tespit etmek amacı ile her bir bölmede kızılötesi yakınlık sensörü kullanılmıştır. Mekanik sistemin başlangıç pozisyonunu (home position) alması ve hareket sınırlarını belirlemek için mekanik anahtarlar tercih edilmiştir.

Bilgisayar yazılımı ise park etmek için gelen aracın fotoğrafını çekip plaka tanımlaması yapmakta ve kaydını tutarak ücret hesaplaması yapmaktadır. Yazılım C# dili ile geliştirlmiş olup Windows tabanlı olarak çalışmaktadır. Elektronik kontrol birimi ile RS-232 protokolünü kullanarak haberleşmektedir. Şekil 5.1'de otomatik otopark sisteminin blok diyagramı görülmektedir.



Şekil 5.1. Otomatik otopark sisteminin blok diyagramı

5.2. Park Bölmelerinin Tasarımı

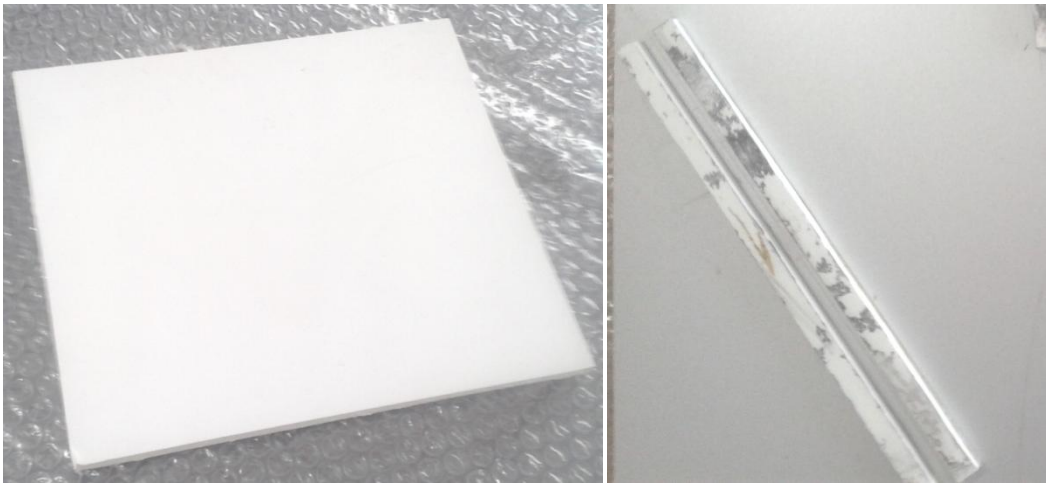
Araçların park etme süresince kalacağı bölmelerdir. Yapılan çalışma prototip niteliğinde olduğu için tasarımda işlemesi kolay, hafif ve dayanıklı plastech PVC foam (yada foreks) levhalardan faydalanılmıştır. Piyasadaki yaygın olarak bilinen adı dekota olduğu için bundan sonra bu tabir kullanılacaktır. Dekota levha Rijit PVC levhanın bir türevidir. Aradaki fark yoğunluğun düşürülmesidir ki, bu işlem toz PVC hammadesinin içerisine ajan adı verilen kimyasallar homojen bir biçimde karıştırılarak ve çekme hattında işlenerek gerçekleştirilir. Malzemenin özgül ağırlığı düşürüldüğü için Rijit PVC'ye göre çok daha ekonomiktir. Standart rengi beyazdır fakat üretim esnasında boya katılarak istenilen renkte üretilebilir. Kalınlıkları 2mm'den 20mm'ye kadar olabilmektedir.

Tasarıma tabandan başlanılmış ve park bölmeleri ile taşıyıcı mekanik sistemi taşıyacak olan en alt tabaka için 5mm kalınlığında alüminyum kompozit levha kullanılmıştır. Bu malzeme oldukça sağlam ve dayanıklıdır.

Resim 5.1.Taşıyıcı alüminyum taban

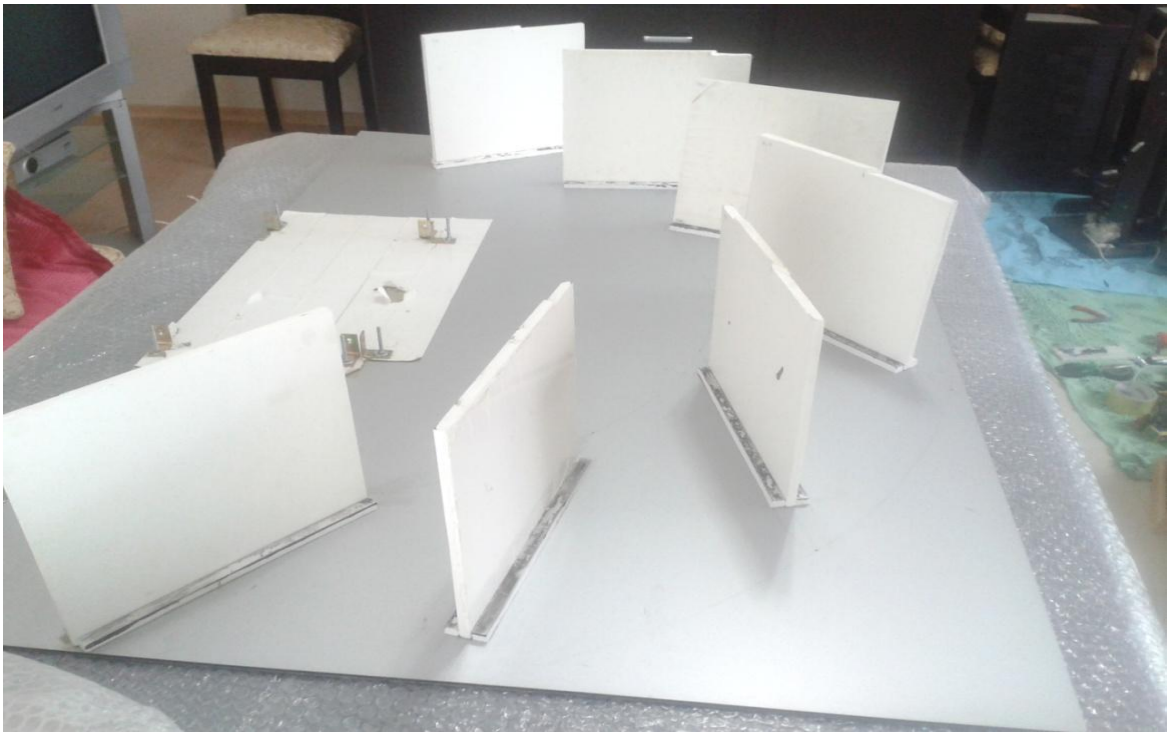
Alüminyum taban üzerine taşıyıcı dikey dekota levhalar yerleştirilecektir. Dikey taşıyıcı dekota levhalar 7mm kalınlığındadır ve 22,5cm x 20cm boyutlarındadır.

Resim 5.2’de de görüldüğü gibi dikey levhaların dik olarak durabilmesi için taban üzerine ince çubuk şeklinde kesilmiş dekotalar sabitlenmiştir.

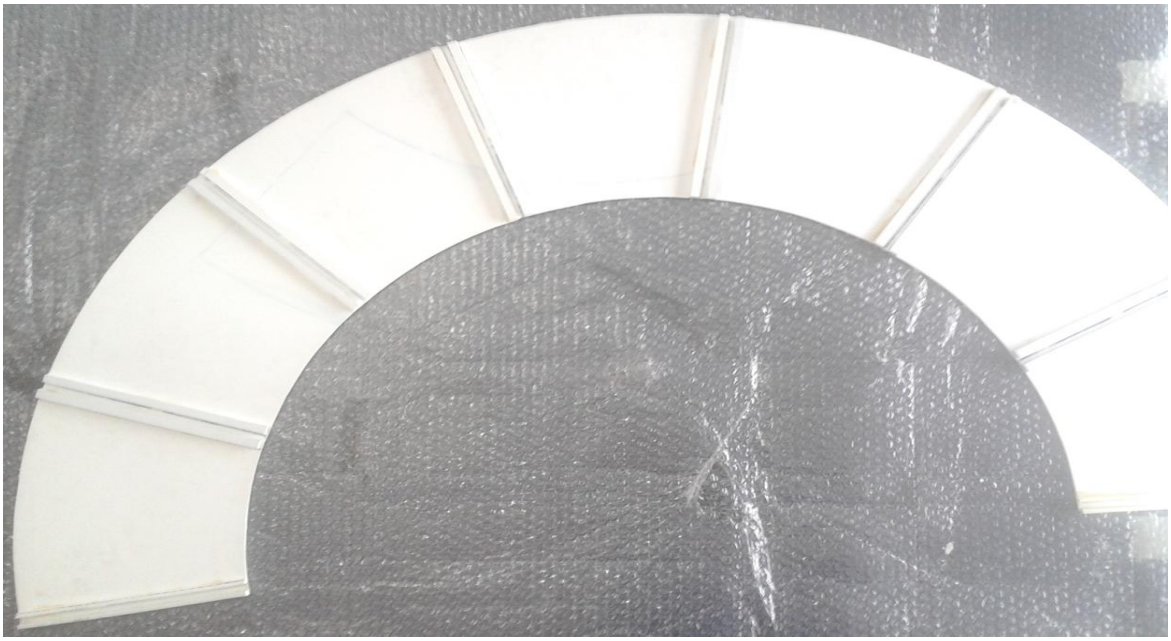


Resim 5.2. Taşıyıcı dekota levha ve yerleştirileceği kanal

Yedi adet dikey dekota levha taban üzerinde bulunan sabit ince çubukların arasına yerleştirildikten sonra otoparkın ilk katını oluşturacak yarım daire şeklindeki 3mm kalınlığında dekota dikey levhaların üzerine oturtulur. Yarım daire şeklindeki dekotalardan 3 adet vardır ve her birinde de dikey dekotaların yerleştirileceği kanallar mevcuttur. Katları oluşturabilmek için yarım daire şeklindeki dekotaların arasına da sekizer adet dikey levhalar konulmuştur. Aşağıdaki resimler (Resim 5.3 – Resim 5.9) incelenirse otoparkın yapısı daha iyi anlaşılacaktır.



Resim 5.3. Dikey dekotaların taban üzerine yerleştirilmesi



Resim 5.4. Park bölmelerinin zeminin oluşturan yarım daire şeklindeki dekota

Resim 5.5. Otaparkın birinci katının zemini

Resim 5.6. Otoparkın ikinci katının bölümlendirilmesi

Resim 5.7. Otoparkın birinci katının tamamlanmış durumu

Resim 5.8. Otoparkın montajının tamamlanmış resmi

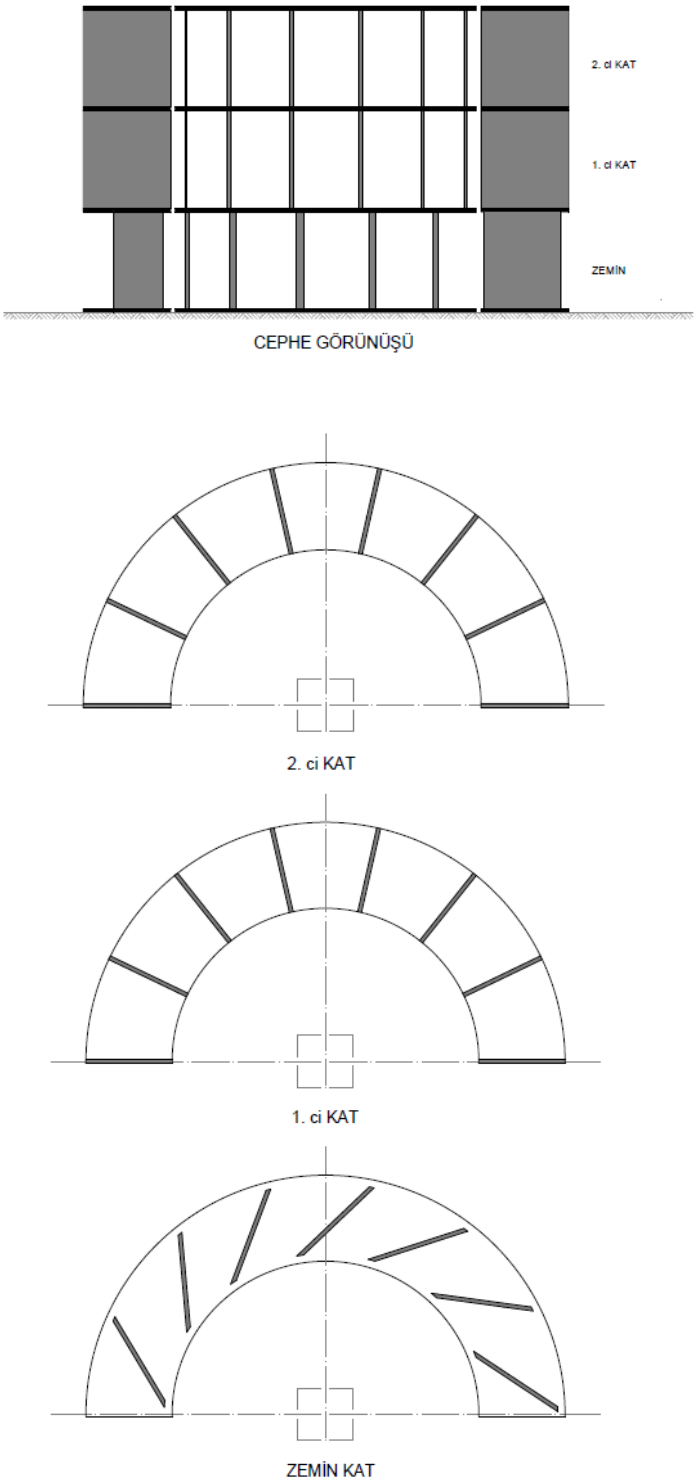
Resim 5.9. Araçların otopark içindeki görüntüsü

Otopark zemin üzerine oturtulmuş iki kattan oluşmaktadır ve her bir katta 7 araçlık park yeri mevcuttur. Toplam 14 araç park edebilir. Fakat birinci kattaki 4. Bölme araç kabul bölmesi olup park etmek için gelen araç buradan işleme alınmakta ve yine teslim edilecek araç buradan teslim edilmektedir. Yani bu prototip uygulamada tasarlanan otopark 13 araca kadar park yeri sağlamaktadır.

Daha öncede bahsedildiği gibi otopark yarım daire şeklinde tasarlanmıştır. Bu da açısal olarak 180^{o}'lik bir açıya karşılık gelir. Her bir katta 7 bölme olduğuna göre her bir araç için $180^{o}/7=25,7^{o}$'lik alan ayrılmıştır. Otoparkın dairesel olarak tasarlanması taşıyıcı robot sisteminin 2 prizmatik 1 döner eksenden oluşan (R2P) bir silindirik koordinatlı robot olmasından kaynaklanmaktadır. Tam dairesel (360^{o}) tasarlanmamasının nedenleri ise prototip niteliğinde olması ve tek bir robotun ihtiyaçlara karşılık vermesinin zor olmasıdır. Gerçekte uygulandığı düşünülürse tek bir taşıyıcı robot isteklere karşılık vermekte yetersiz kalacak ve kuyruk oluşacaktır. Her biri 180^{o}'lik alanlardan sorumlu iki robot ile müşterilerin hem istekleri daha hızlı karşılanabilecek hem de ticari olarak daha fazla kazanç elde edilebilecektir.

Otoparkın birinci katı "A Katı", ikinci katı ise "B Katı" olarak isimlendirilmiştir. A katındaki bölmeler soldan sağa doğru A1, A2, A3, A4, A5, A6, A7 olarak, B katındaki bölmeler ise yine soldan sağa doğru B1, B2, B3, B4, B5, B6, B7 olarak

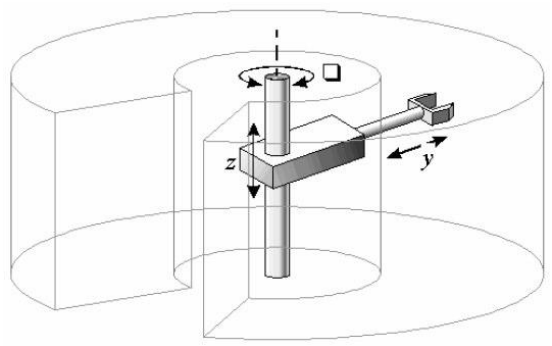
isimlendirilmişlerdir. A4 bölmesi araç kabul ve teslim bölmesi olup daha önce de bahsedildiği gibi araç parkı için kullanılmamaktadır. Otoparka ilişkin teknik çizimler Şekil 5.2'de görülebilir.



Şekil 5.2. Otoparkın bilgisayar ortamında çizilmiş cephe görünüşü ve kat planları

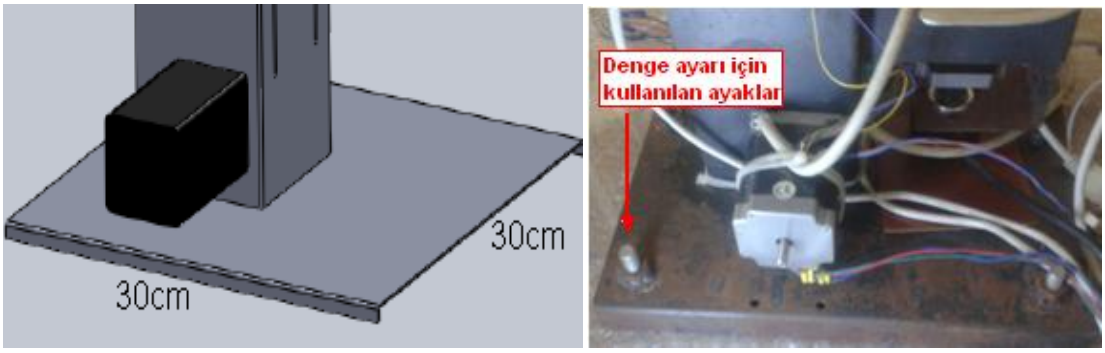
5.3. Taşıyıcı Elektromekanik Sistemin Tasarımı

Çalışmanın en kompleks ve en hassas bölümüdür. Burada kullanılan yapı bölüm girişinde de bahsedildiği gibi bir robot sistemidir. İş yüklenecek robotun gerçekleştirmesi gereken görevler göz önüne alındığında iki prizmatik (Y ve Z eksenleri) ve bir döner (X ekseni) eksende hareket eden robot amaca uygundur. Manipülatör yapısına göre bu robot R2P olarak isimlendirilir ve silindirik koordinatlı robottur. Şekil 5.3 silindirik koordinatlı bir robotun manevra alanını göstermektedir.



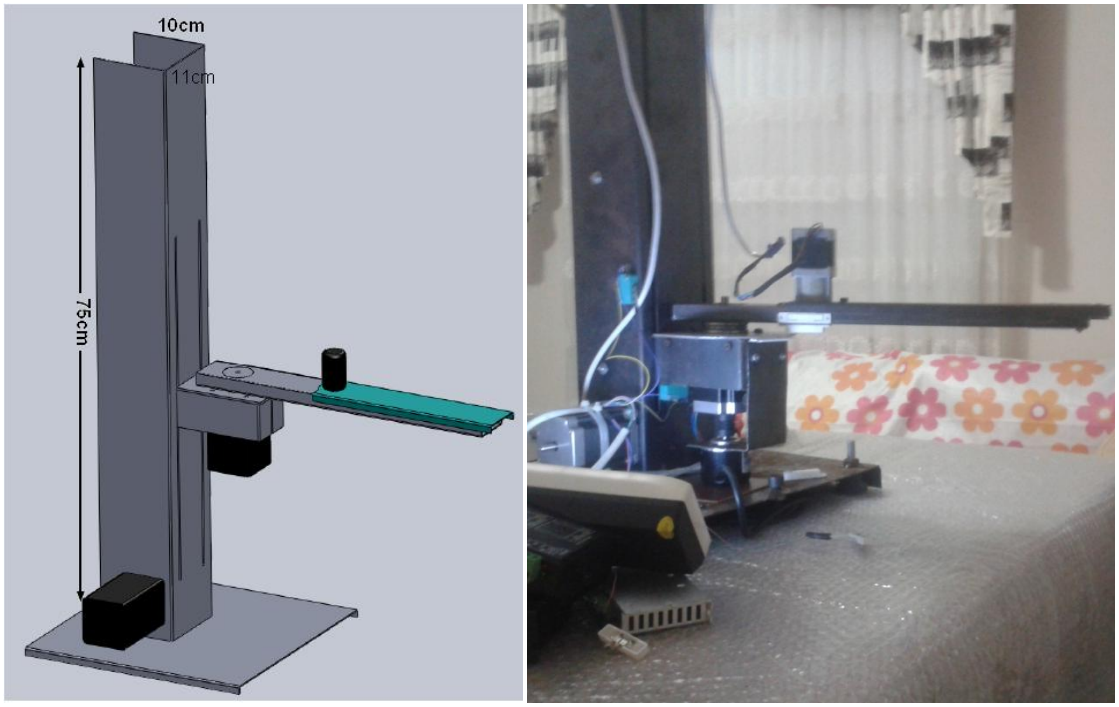
Şekil 5.3. Silindirik koordinatlı bir robot ve çalışma alanı [23]

Taşıyıcı elektromekanik sistem 1,5mm sacdan yapılmış ve adım motor, dişliler, trigger kayışı, raylar, eksenel-sabit bilyalı rulman gibi bileşenlerden oluşmaktadır. Tasarıma tabandan başlanmış ve kenarlarından 3cm bükülerek yerden yükseltilen ve 30cm X 30cm ebatlarında sac bir taban elde edilmiştir. Bu taban elektromekanik sistemin tüm yükünü taşıyacaktır. Ayrıca oluşacak dengesizlikleri ayarlamak için tabanın 4 köşesine ayarlı ayaklar monte edilmiştir. Olası bir dengesizlik sistemin stabil çalışmasını etkileyeceğinden bu ayarlı ayaklar vasıtası ile dengesizlik giderilebilecektir (Resim 5.10).

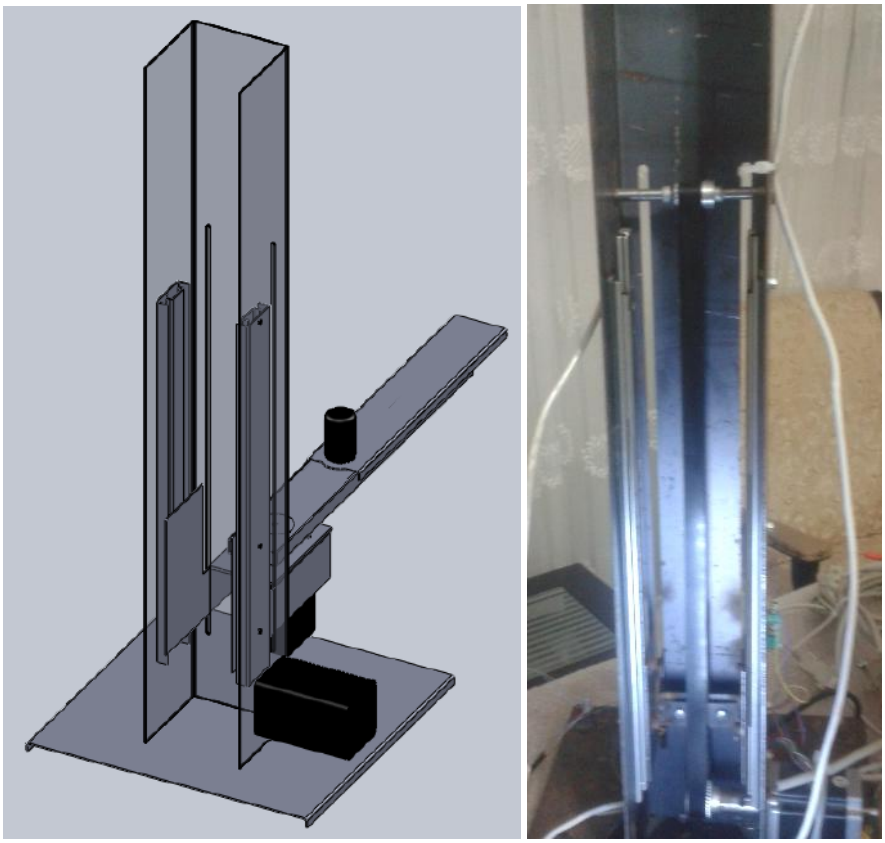


Resim 5.10. Elektromekanik sistemin tabanı

Tabanın üzerine 75x10x11cm boyutlarında bir kule kaynak yapılmak suretiyle sabitlenmiştir. Bu kulenin arkaya bakan yüzeyi yoktur. Buradan kulenin iç kısmına dikey hareket için gerekli kayış-kasnak mekanizması ve raylar yerleştirilecektir. Ayrıca kuleye karşıdan bakılırsa sol tarafında dikey hareketi sağlayan adım motorun sabitlendiği görülür. Kulenin ön yüzünde ise birbirine paralel 48cm boyunda iki dikey ince boşluk görülür. Bu boşluklar ön tarafta yer alan, yatay ve ileri/geri hareketi sağlayan diğer parçaları taşıyan kısmın dikey hareket esnasında taşınması için açılmış boşluklardır (Resim 5.11 ve Resim 5.12).

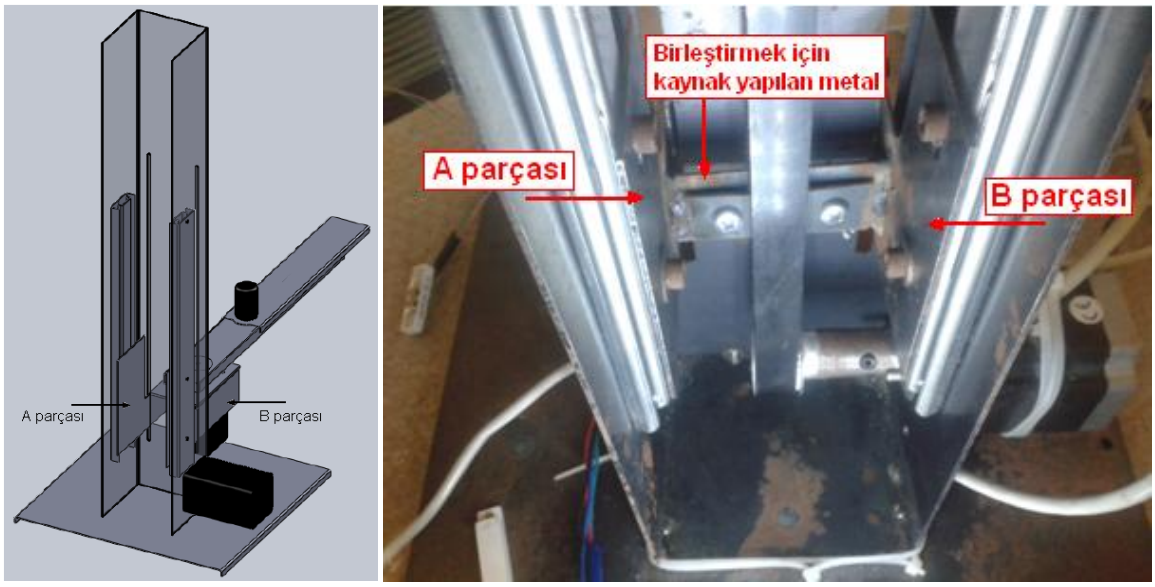


Resim 5.11. Kulenin önden görünüşü



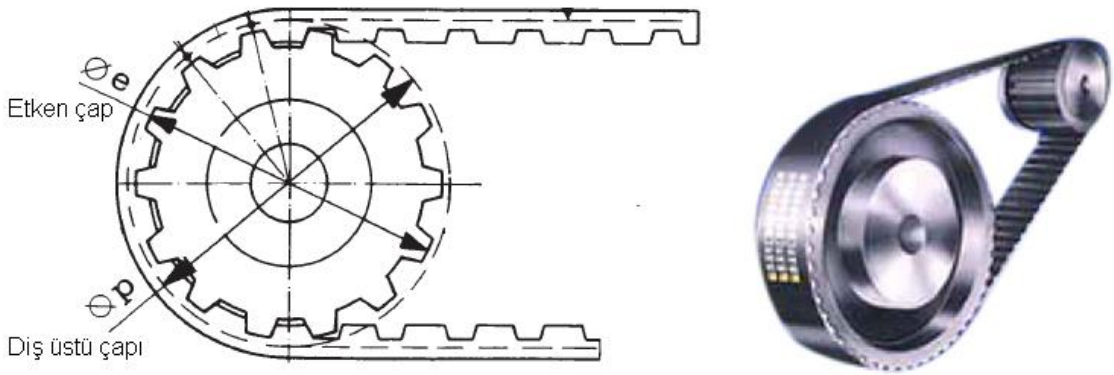
Resim 5.12. Kulenin arkadan görünüşü

Taban üzerindeki kulenin iç yan kısımlarına birer adet çekmece rayı sabitlenmiştir. Bu raylar dikey hareketin sağlanması içindir. Dikey hareket esnasında ön kısımda yer alan diğer mekanizmalar (yatay ve ileri/geri hareketi sağlayan) yardımcı bileşenler vasıtasıyla bu raylara bağlanacaktır. Daha sonra Resim 5.13'de görülen A ve B olarak isimlendirilen iki metal parça bu raylara simetrik olarak sabitlenmiş ve aralarına konulan başka bir metal parça ile kaynak yapılarak birleştirilmişlerdir. Artık her iki raya bağlı A ve B metal parçaları birlikte hareket edebileceklerdir. A ve B metal parçalarının bir kısmı kulenin iç tarafında kalırken bir kısmı da ön tarafından çıkmaktadır.

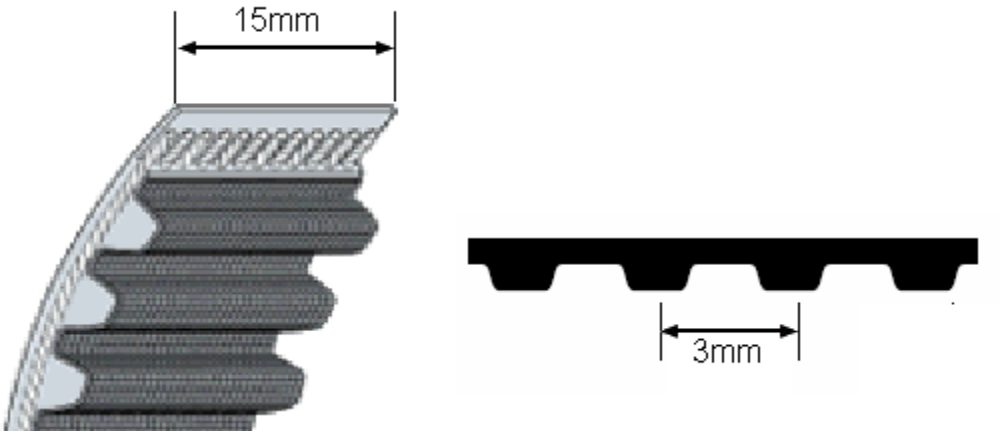


Resim 5.13. Ray üzerine sabitlenmiş A ve B parçaları

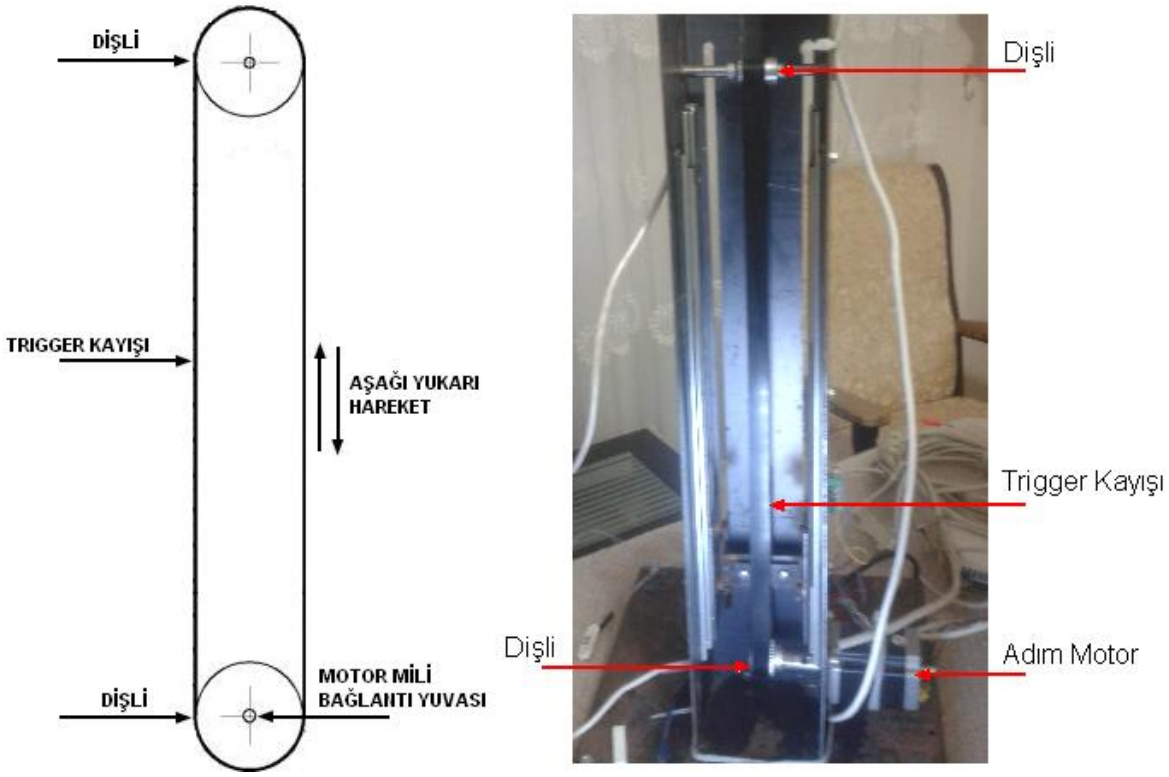
Dikey hareketi sağlayacak adım motor kulenin soluna sabitlenmiş ve ucuna diş sayısı 30 olan düz bir dişli monte edilmiştir. Dişlinin diş üstü çapı 28,65mm, etken çapı ise 25,98mm'dir. Karşılık olarak üst taraf ise diş sayısı 16, diş üstü çapı 15,28mm ve etken çapı 14,52mm olan başka bir dişli monte edilerek bu iki dişli birbirine 3m15'lik bir trigger kayışı ile bağlanmıştır. 3m15, kayış genişliğinin 15mm bir diş merkezinden komşu diğer diş merkezine mesafenin 3mm olduğunu ifade etmektedir. Dişli merkezler arasındaki mesafe 40cm'dir. Trigger kayışının bir tarafı daha önce bahsedilen A ve B parçalarını birbirine kaynak yolu ile birleştiren parçaya küçük bir metal ile sabitlenmiştir. Böylece adım motor dönme hareketi yaptığında kayış ile beraber ön taraftaki X ve Y eksenlerinde hareketi sağlayacak mekanizmalar yukarı/aşağı hareket ederek dikey hareket sağlanmış olacaktır.



Şekil 5.4. Dişli ve kayış



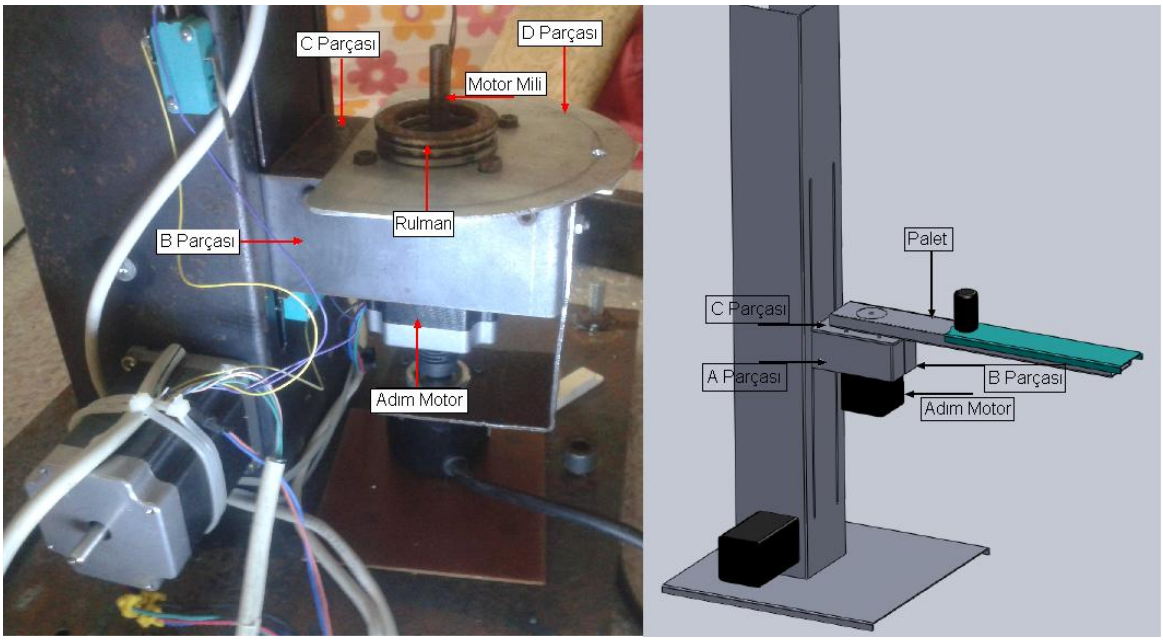
Şekil 5.5. Kullanılan trigger kayışının ölçüleri



Şekil 5.6. Kayış ve dişli sisteminin görünüşü

Yatay hareketi sağlayacak mekanizma kulenin ön tarafındadır. Daha önce sabitlenen A ve B parçalarına C parçası da monte edilerek yatay hareket için gerekli altyapı sağlanmış olur. Uygun montaj delikleri ve motor milinin geçmesi için uygun yuvanın açıldığı C parçasına, dikey hareketi sağlayan adım motora eş başka bir adım motor sabitlenmiştir. Ayrıca C parçasının üstüne yüzeyi genişletmek için alüminyum D parçası vidalanmıştır.

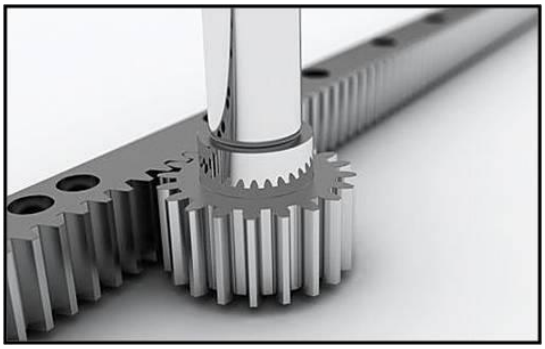
Yatayda hareket edecek palet sisteminin motor miline bağlantısı yapılmadan önce altına eksenel-sabit bilyalı rulman konulmuştur. Rulmanın kullanılma amacı yatay silindirik harekette sürtünmeyi azaltmak ve yükün tamamen motor miline binmesini önlemektir. Yatay hareketi (X ekseni) sağlayan elemanlar Resim 5.14'de görülmektedir.



Resim 5.14. Yatay hareketi sağlayan elemanlar

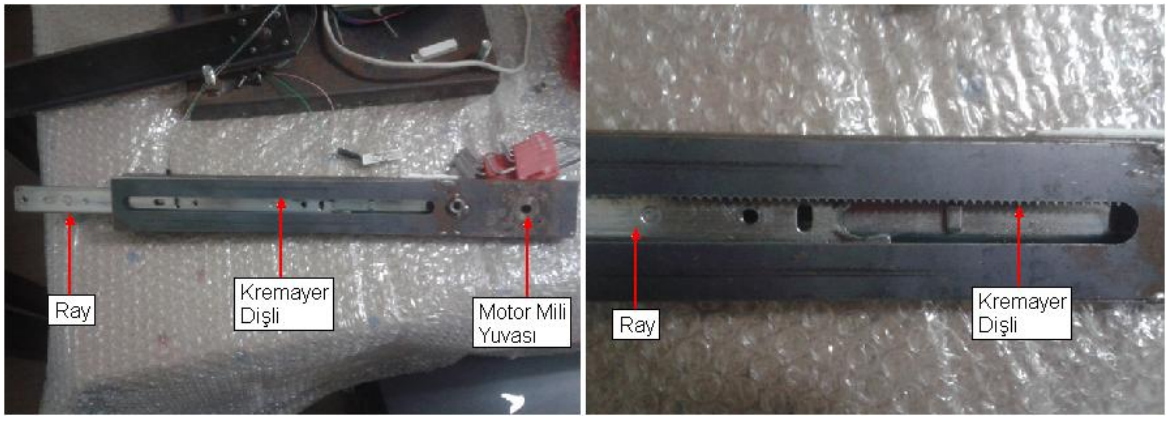
Yatay hareketi sağlayan motor miline palet sistemi monte edildiğinde elektromekanik sistem hazır duruma gelecektir. Palet hem yatayda 180^0'lik bir alanda hareketi hem de ileri/geri (Y ekseni) hareketi sağlamaktadır. Yatay hareket, şekilden de anlaşılacağı üzere motor milinin hareket etmesi ile sağlanmaktadır. İleri/geri hareket ise palet sisteminde kullanılan daha düşük torklu bir adım motor ve kremayer dişli sistemi vasıtası ile gerçekleşmektedir.

Kremayer dişli üst yüzünde dişler bulunan doğrusal bir çubuk olarak düşünülebilir. Kremayer dişliler dairesel hareketin doğrusal harekete çevrilmesi amacıyla kullanılır. Kremayer dişli düz bir yüzeye açılan dişler sayesinde, dönebilen bir dişlinin hareketini tek yönde ilerlemeye çevirir. Kremayer dişlinin üzerinde dönen dişliye pinyon adı verilir. Resim 5.15 incelenirse kremayer dişlinin mantığı daha iyi anlaşılacaktır.



Resim 5.15. Kremayer dişli ve pinyon dişli

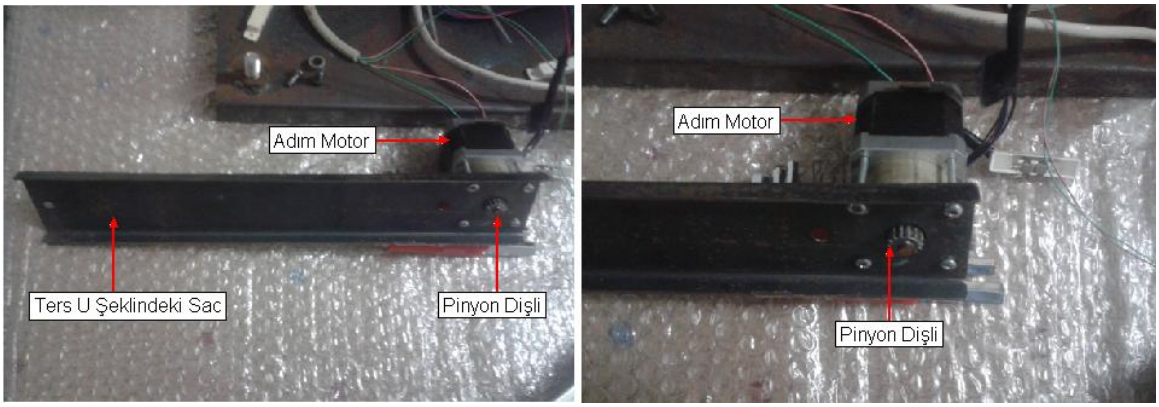
Palet sistemi yapılırken 40x8cm ebatlarındaki metal sac plaka uzun kenarlarından 1,5cm bükülerek U şeklinde, 40x5x1,5cm'lik bir oluk yapılmıştır. Oluk yapılmasındaki amaç ileri geri hareket için kullanılacak olan çekmece rayının buraya yerleştirilecek olmasıdır. Sac U şekline getirildikten sonra yatay motor mili için ve diğer elemanların montajı için uygun yerlere delikler açılmıştır. Daha sonra oluğun içerisine çekmece rayı vidalanarak sabitlenmesi sağlanır. Rayın sabitlendiği oluğun üzerine kremayer dişliyi barındıran sac, kaynak yapılmak suretiyle sabitlenmiştir (Resim 5.16).



Resim 5.16. Palet sistemindeki kremayer dişli ve ray

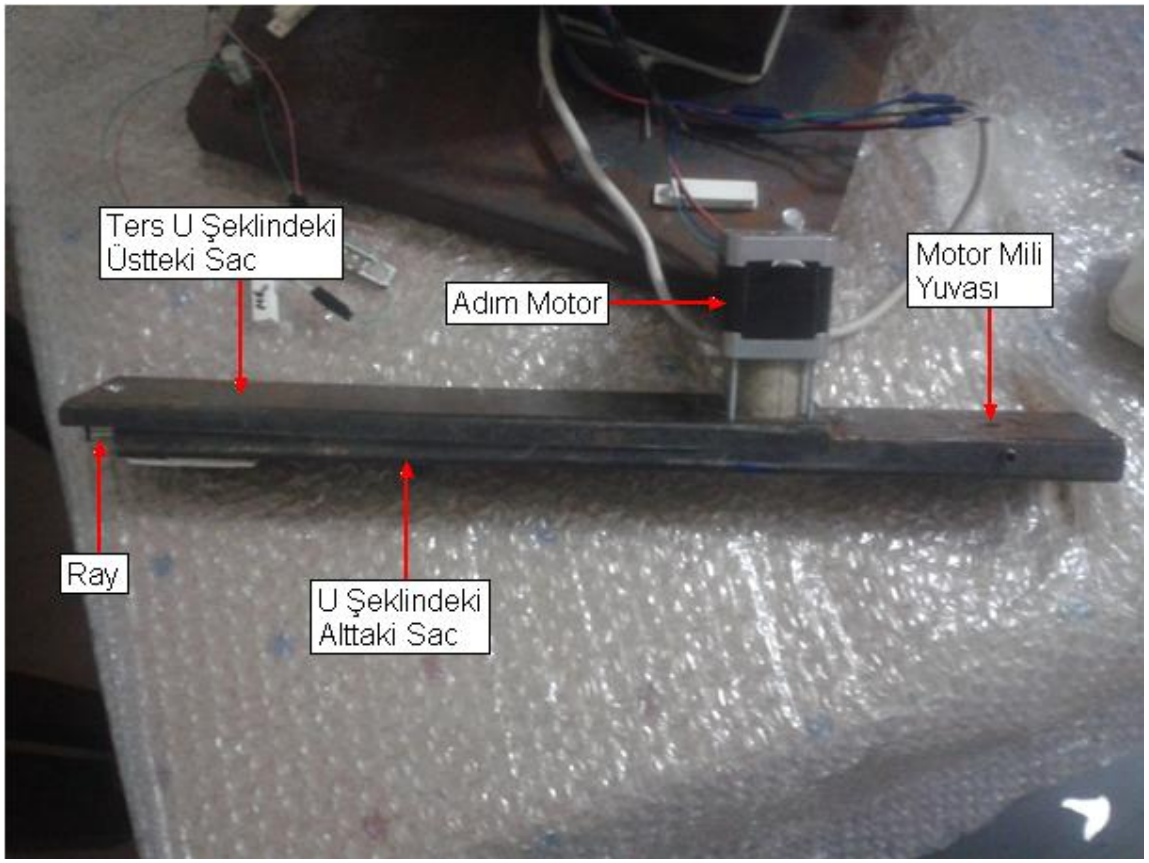
Paletin ikinci önemli bileşeni ise kremayer dişli üzerinde ileri/geri hareketi sağlayacak olan parçadır. 30x8cm ebatlarındaki bir metal sac uzun kenarlarından 1cm bükülerek ters U şeklinde 30x6x1cm ebatlarında başka bir oluk elde edilmiştir. Bu parçanın üzerine 0,5 Nm tork üretebilen bir adım motor yerleştirilmiş ve motor miline pinyon adı verilen dişli sabitlenmiştir. Son olarak adım motorun bulunduğu bu parça diğer parçanın üzerine yerleştirilerek uç tarafı çekmece rayının hareketli kısmına vida vasıtası ile bağlanmıştır. Pinyon dişlisi kremayer dişli

üzerinde hareket ederek, ray yardımı ile, üstteki parçanın ileri geri hareketini sağlamaktadır (Resim 5.17).

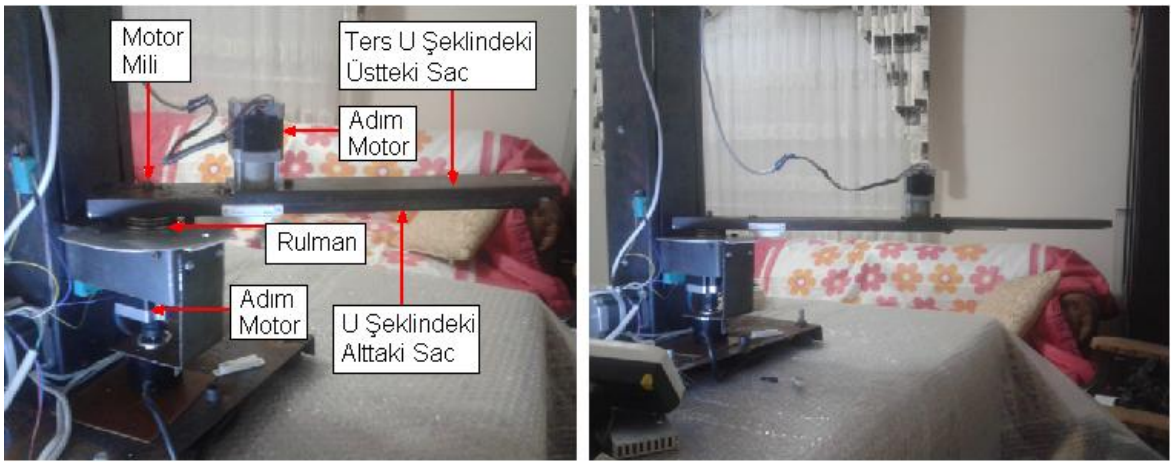


Resim 5.17. Palet sistemindeki pinyon dişli ve adım motor

Palet sistemi motor mili yuvasından yatay hareketi sağlayan motor mili üzerine yerleştirilerek mekanizma tamamlanmıştır. Resim 5.18 ve Resim 5.19 incelenerek palet sisteminin çalışması daha iyi anlaşılabilir.



Resim 5.18. Palet sisteminin birleştirilmiş görünüşü



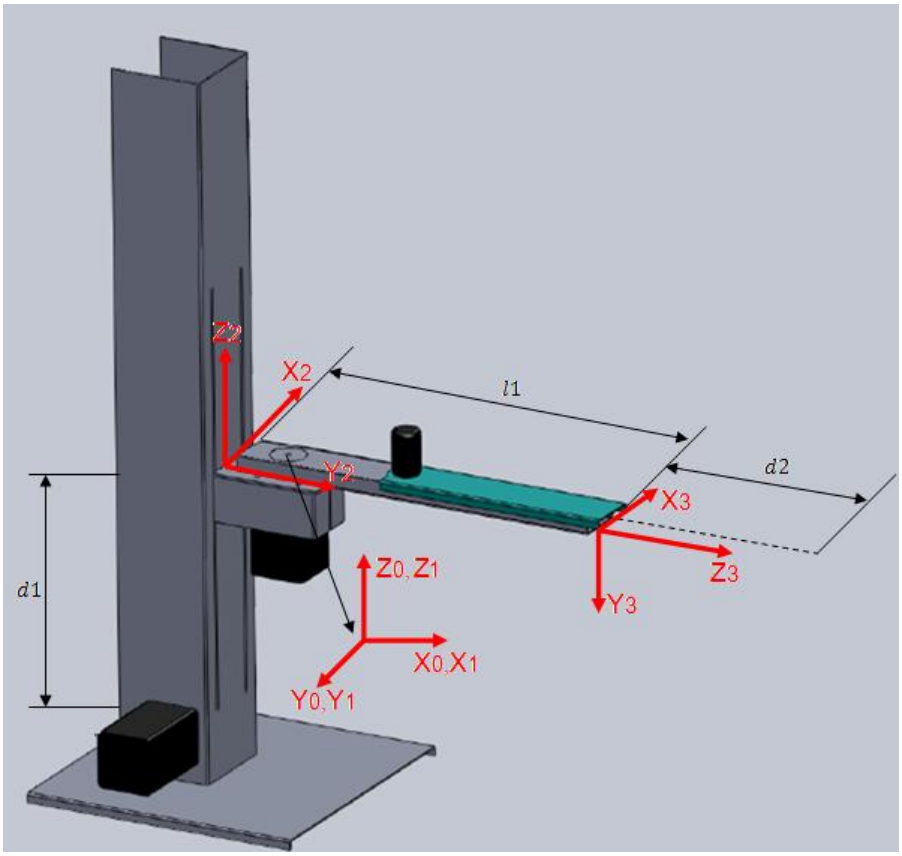
Resim 5.19. Palet sisteminin taşıyıcı sistem üzerindeki görüntüsü

5.3.1. Elektromekanik robot sisteminin ileri kinematik hesaplamaları

İleri kinematik; her eklemin sonraki ekleme göre açısının veya konumunun verilerek uç işlevcinin konumunun hesaplanması (bulunması) işlemidir. Bu çalışmada, kinematik hesaplamalar için Denavit-Hartenberg (D-H) yöntemi kullanılmıştır. D-H tablosunun oluşturulması amacı ile eksenler Şekil 5.7'de gösterildiği gibi yerleştirilmiştir. Daha sonra sırasıyla aşağıdaki işlemler gerçekleştirilmiştir [34, 35]:

1. Sıfırıncı eksenle birinci eksen çakışık alınmıştır. Ekseni yerleştirirken, dönme ve doğrusal hareket yönleri Z ekseni üzerinde olacak şekilde yerleştirilmiştir.
2. Z eksenlerini yerleştirdikten sonra X eksenleri yerleştirilmiştir. X eksenini yerleştirirken sonraki Z ekseni göz önünde bulundurularak; X eksenini kendi ekseninde döndürdüğümüzde sonraki Z ekseninin doğrusal veya dönme hareketinin olduğu eksene denk gelmesi göz önüne alınmıştır.
3. Eksenleri yerleştirdikten sonra X ve Z eksenleri arasındaki uzaklık ve açıları Çizelge 5.1'deki D-H tablosuna yazılmıştır.

Robot sisteminin her bir eklemi için yukarıdaki açıklamalara uygun olarak yerleştirmeleri yapılarak D-H tablosu Çizelge 5.1'de verilen şekilde çıkarılmıştır. Burada, α_{i-1}, Z ekseni ile bir önceki Z ekseni arasındaki açı, a_{i-1}, Z ekseni ile bir önceki Z ekseni arasındaki uzaklık, d_i, X ekseni ile bir önceki X ekseni arasındaki uzaklık, θ_i, X ekseni ile bir önceki X ekseni arasındaki açıdır.



Şekil 5.7. Elektromekanik taşıyıcı robotun eksen sistemi

Çizelge 5.1. D-H tablosu

i	α_{i-1}	a_{i-1}	d_i	θ_i	Değişken
1	0	0	0	θ_1	θ_1
2	0	0	d_1	0	d_1
3	-90	0	$d_2 + l_1$	0	d_2

Her eklem için dönüşüm matrislerinin çıkarılmasında Eş. 5.1'de verilen dönüşüm matrisi kullanılmıştır.

$$ {}^{i-1}_{i}T = \begin{matrix} cos\theta_i & -sin\theta_i & 0 & a_{i-1} \\ sin\theta_i cos\alpha_{i-1} & cos\theta_i cos\alpha_{i-1} & -sin\alpha_{i-1} & -sin\alpha_{i-1} d_i \\ sin\theta_i sin\alpha_{i-1} & cos\theta_i sin\alpha_{i-1} & cos\alpha_{i-1} & cos\alpha_{i-1} d_i \\ 0 & 0 & 0 & 1 \end{matrix} \quad (5.1) $$

İlgili eksenlere göre elde edilen dönüşüm matrisleri sırası ile Eş. 5.2, 5.3 ve 5.4'de verilmiştir.

$$ {}^{0}_{1}T = \begin{matrix} cos\theta_1 & -sin\theta_1 & 0 & 0 \\ sin\theta_1 & cos\theta_1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{matrix} \tag{5.2} $$

$$ {}^{1}_{2}T = \begin{matrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & d_1 \\ 0 & 0 & 0 & 1 \end{matrix} \tag{5.3} $$

$$ {}^{2}_{3}T = \begin{matrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & d_2 + l_1 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{matrix} \tag{5.4} $$

Bütün eklemlerin dönüşüm matrisini çıkardıktan sonra her bir eklemin dönüşüm matrisleri birbirleriyle çarpılmış ve Eş. 5.5'de verilen ileri kinematik dönüşüm matrisi elde edilmiştir.

$$ {}^{0}_{3}T = {}^{0}_{1}T\,{}^{1}_{2}T\,{}^{2}_{3}T = \begin{matrix} cos\theta_1 & -sin\theta_1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ sin\theta_1 & cos\theta_1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & d_2 + l_1 \\ 0 & 0 & 1 & 0 & 0 & 0 & 1 & d_1 & 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \end{matrix} \tag{5.5} $$

$$ = \begin{matrix} cos\theta_1 & 0 & -sin\theta_1 & -sin\theta_1(d_2 + l_1) \\ sin\theta_1 & 0 & cos\theta_1 & cos\theta_1(d_2 + l_1) \\ 0 & -1 & 0 & d_1 \\ 0 & 0 & 0 & 1 \end{matrix} $$

5.3.2. Elektromekanik robot sisteminin ters kinematik hesaplamaları

Ters kinematik, uç işlevcinin konumuna göre her eklemin konum ve açısal değerlerinin hesaplanması (bulunması) işlemidir. Ters kinematik matrisini bulurken analitik çözüm yaklaşımı kullanılır. Her iki tarafı birinci eklemin dönüşüm matrisinin tersini ($[{}^{0}_{1}T]^{-1}$) bulup çarptığımızda Eş. 5.6 elde edilir [34, 35]:

$$ [{}^{0}_{1}T]^{-1}\,{}^{3}_{0}T = [{}^{0}_{1}T]^{-1}\,{}^{0}_{1}T\,{}^{1}_{2}T\,{}^{2}_{3}T \tag{5.6} $$

Eş. 5,6'da $[{}^{0}_{1}T]^{-1}\,{}^{0}_{1}T = 1$ olduğundan ilgili denklem aşağıdaki gibi yazılır.

$$[{}^{0}_{1}T]^{-1}\,{}^{3}_{0}T = {}^{1}_{2}T\,{}^{2}_{3}T \tag{5.7}$$

Dönüşüm matrisinin tersini bulmak için Eş. 5.8 göz önüne alınmıştır. Burada R dönme matrisi P ise konum vektörüdür.

$$[{}^{i-1}_{i}T]^{-1} = \begin{matrix} [{}^{i-1}_{i}R]^{T} & -[{}^{i-1}_{i}R]^{T}\ {}^{i-1}_{i}P \\ 000 & 1 \end{matrix} \tag{5.8}$$

Bu denklem, birinci eklem için uygulanarak Eş. 5.9 bulunmuştur.

$$[{}^{0}_{1}T]^{-1} = \begin{matrix} [{}^{0}_{1}R]^{T} & -[{}^{0}_{1}R]^{T}\ {}^{0}_{1}P \\ 000 & 1 \end{matrix} \tag{5.9}$$

Birinci eklemin dönüşüm matrisinden dönme matrisi kısmı alınmış ve

$${}^{0}_{1}R = \begin{matrix} cos\theta_1 & -sin\theta_1 & 0 \\ sin\theta_1 & cos\theta_1 & 0 \\ 0 & 0 & 1 \end{matrix} \tag{5.10}$$

olarak bulunmuştur.

Daha sonra aşağıda verilen işlemler izlenerek Eş. 5.13 ile verilen dönüşüm matrisinin tersi elde edilmiştir.

$$[{}^{0}_{1}R]^{T} = \begin{matrix} cos\theta_1 & sin\theta_1 & 0 \\ -sin\theta_1 & cos\theta_1 & 0 \\ 0 & 0 & 1 \end{matrix} \tag{5.11}$$

$$-[{}^{0}_{1}R]^{T}\,{}^{0}_{1}P = -\begin{matrix} cos\theta_1 & sin\theta_1 & 0 & 0 \\ -sin\theta_1 & cos\theta_1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{matrix} = \begin{matrix} 0 \\ 0 \\ 0 \end{matrix} \tag{5.12}$$

$$[{}^{0}_{1}T]^{-1} = \begin{matrix} cos\theta_1 & sin\theta_1 & 0 & 0 \\ -sin\theta_1 & cos\theta_1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{matrix} \tag{5.13}$$

Eş. 5.13, Eş. 5.14'de kullanılmış ve ${}^{3}_{1}T$ matrisi elde edilmiştir.

$$[{}^{0}_{1}T]^{-1}\,{}^{3}_{0}T = {}^{3}_{1}T = \begin{matrix} cos\theta_1 & sin\theta_1 & 0 & 0 & r_{11} & r_{12} & r_{13} & p_x \\ -sin\theta_1 & cos\theta_1 & 0 & 0 & r_{21} & r_{22} & r_{23} & p_y \\ 0 & 0 & 1 & 0 & r_{31} & r_{32} & r_{33} & p_z \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \end{matrix} \tag{5.14}$$

$$= \begin{matrix} r_{11}cos\theta_1 + r_{21}sin\theta_1 & r_{12}cos\theta_1 + r_{22}sin\theta_1 & r_{13}cos\theta_1 + r_{23}sin\theta_1 & p_xcos\theta_1 + p_ysin\theta_1 \\ r_{21}cos\theta_1 - r_{11}sin\theta_1 & r_{22}cos\theta_1 - r_{12}sin\theta_1 & r_{23}cos\theta_1 - r_{13}sin\theta_1 & p_ycos\theta_1 - p_xsin\theta_1 \\ r_{31} & r_{32} & r_{33} & p_z \\ 0 & 0 & 0 & 1 \end{matrix}$$

Daha sonra ${}^{3}_{1}T = {}^{1}_{2}T\ {}^{2}_{3}T$ eşitliğinin kullanılması ile yine ${}^{3}_{1}T$ matrisi eşitlk 5.16'da gösterildiği şekilde elde edilmiştir.

$${}^{3}_{1}T = \begin{matrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & d_1 \\ 0 & 0 & 0 & 1 \end{matrix} \begin{matrix} 1 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 1 & 0 & d_2 + l_1 \\ 0 & 0 & 0 & 1 \end{matrix} = \begin{matrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & d_2 + l_1 \\ 0 & -1 & 0 & d_1 \\ 0 & 0 & 0 & 1 \end{matrix} \tag{5.15}$$

$$\begin{matrix} r_{11}cos\theta_1 + r_{21}sin\theta_1 & r_{12}cos\theta_1 + r_{22}sin\theta_1 & r_{13}cos\theta_1 + r_{23}sin\theta_1 & p_xcos\theta_1 + p_ysin\theta_1 \\ r_{21}cos\theta_1 - r_{11}sin\theta_1 & r_{22}cos\theta_1 - r_{12}sin\theta_1 & r_{23}cos\theta_1 - r_{13}sin\theta_1 & p_ycos\theta_1 - p_xsin\theta_1 \\ r_{31} & r_{32} & r_{33} & p_z \\ 0 & 0 & 0 & 1 \end{matrix}$$

$$= \begin{matrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & d_2 + l_1 \\ 0 & -1 & 0 & d_1 \\ 0 & 0 & 0 & 1 \end{matrix} \tag{5.16}$$

Eş. 5.14 ve 5.16'da 4. sütunları (konum vektörü) birbirine eşitlenerek robotun değişkenlerinin uç işlevcinin konumuna göre değerleri elde edilmiştir. İlgili hesaplamalar aşağıda verilmiştir:

$$p_x \cos\theta_1 + p_y \sin\theta_1 = 0 \tag{5.17}$$

$$p_x \cos\theta_1 = -p_y \sin\theta_1 \tag{5.18}$$

$$p_x = \frac{-p_y \sin\theta_1}{\cos\theta_1} \tag{5.19}$$

$$\tan\theta_1 = \frac{p_x}{-p_y} \tag{5.20}$$

$$\theta_1 = \tan^{-1} \frac{p_x}{-p_y} \quad (5.21)$$

$$p_y \cos\theta_1 - p_x \sin\theta_1 = d_2 + l_1 \quad (5.22)$$

$$\frac{p_y \cos\theta_1{}^2 + p_y \sin\theta_1{}^2}{\cos\theta_1} = \frac{p_y}{\cos\theta_1} = d_2 + l_1 \quad (5.23)$$

$$p_y = d_2 + l_1 \cos\theta_1 \quad (5.24)$$

$$p_x \cos\theta_1 + p_y \sin\theta_1 = 0 \quad (5.25)$$

$$p_x \cos\theta_1 + d_2 + l_1 \cos\theta_1 \sin\theta_1 = 0 \quad (5.26)$$

$$p_x \cos\theta_1 = -\ d_2 + l_1 \cos\theta_1 \sin\theta_1 \quad (5.27)$$

$$p_x = -\ d_2 + l_1 \sin\theta_1 \quad (5.28)$$

$$d_2 = \frac{p_x}{\sin \tan^{-1} \frac{p_x}{-p_y}} - l_1 \quad (5.29)$$

Sonuç olarak,

$$p_z = d_1 \quad (5.30)$$

$$d_2 = \frac{p_x}{\sin \tan^{-1} \frac{p_x}{-p_y}} - l_1 \quad (5.31)$$

$$\theta_1 = \tan^{-1} \frac{p_x}{-p_y} \quad (5.21)$$

olarak bulunmuştur.

5.3.3. Elektromekanik sistemde kullanılan tahrik elemanlarının seçimi

Hatırlanacağı üzere taşıyıcı elektromekanik sistem R2P olarak isimlendirilen silindirik koordinatlı bir robottur. Bu robot sisteminde 3 adet adım motor

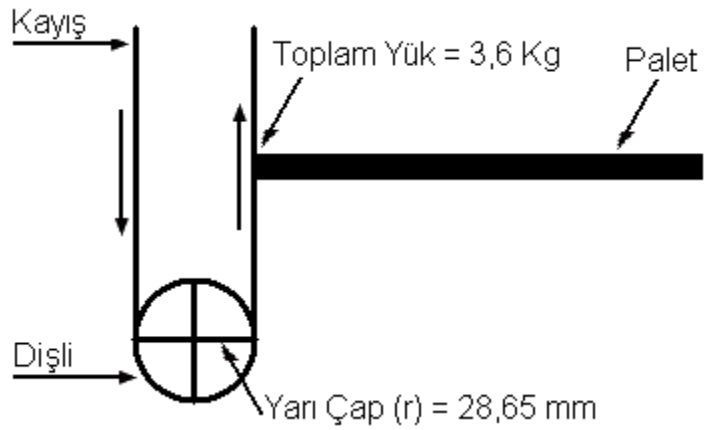
kullanılmıştır. Kullanılan adım motorlar SOYO marka olup dikey ve yatay hareketler için 1,85Nm tork üretebilen, ileri geri hareket için ise 0,5Nm tork üretebilen motorlar kullanılmıştır.

Adım motoru, genel olarak elektrik enerjisini istenilen açıda dönme hareketine çevirebilen elektro-mekanik bir cihaz olarak tanımlanabilir. Adım motorlar ile ilgili bilgi üçüncü bölümde detaylı olarak verilmiştir.

Kullanılan adım motorların ilki dikey hareketi sağlayan kayış kasnak sistemini hareket ettiren adım motordur. Çizelge 5.2'de dikey hareket esnasında motora binen parçaların ağırlıkları verilmiş ve daha sonra hesap yapılarak motorun sahip olması gereken minimum tork değeri belirlenmiştir.

Çizelge 5.2. Dikey adım motora binen parça ağırlıkları

Dikey adım motora binen parça ağırlıkları	
Yatay hareketi sağlayan adım motor	1000 gr
İleri/geri hareketi sağlayan adım motor	500 gr
A parçası	150 gr
B parçası	150 gr
C Parçası	100 gr
Rulman	100 gr
Palet Sistemi	1500 gr
Taşınacak araba	100 gr
Toplam	3600 gr



Şekil 5.8. Dişliye bağlı kayışın taşıyacağı toplam yük

Çizelge 5.6'deki ağırlıklar ve Şekil 5.8'e göre motor mili ile beraber dönen dişliye bağlı trigger kayışına binen yük, yerçekimi ivmesi ve dişli yarıçapı dikkate alınmış; sürtünmeler önemsenmemiştir.

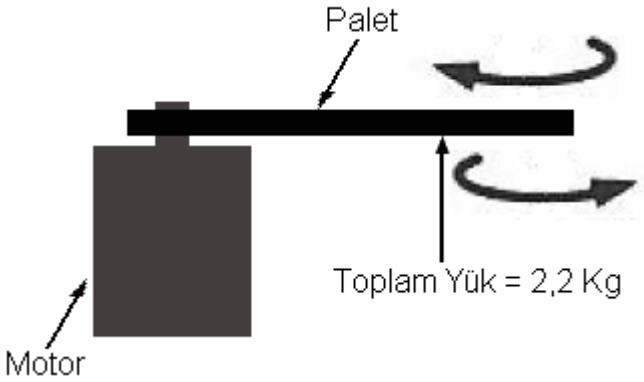
$$G = m.g \tag{5.32}$$
$$G = 3{,}6Kg.9{,}81\ {}^{m}_{sn^2}$$
$$G = 35{,}316N$$

$$T\ Tork\ = G.l \tag{5.33}$$
$$T\ Tork\ = 35{,}316\,.0{,}02865$$
$$T\ Tork\ = 1{,}01\,Nm$$

Bu hesaplamada ;
m; kütle (kg)
g; yerçekimi ivmesi m/sn^2
G; ağırlık (N)
l; uzunluk (m), dişli yarı çapı
T; tork (Nm)

değerlerini ifade etmektedir. Hesaplamalar sonucunda kullanılacak adım motorun minimum 1.01Nm'lik bir tork değerine sahip olması gerektiği anlaşılmaktadır. Burada sürtünmeler için %25 tolerans eklenirse minimum tork değeri yaklaşık 1.26Nm olmalıdır. Motorun özelliklerinin verildiği Çizelge 5.3 incelendiğinde dikey hareket için seçilen motorun uygun olduğu anlaşılmaktadır.

Yatay hareketi sağlayan motorun sahip olması gereken dönme atalet kuvvet (rotor ataleti) değeri hesaplanırken yapılan hareketin çizgisel değil açısal olduğu göz önüne alınmıştır. Palet sistemi için gereli minimum dönme atalet kuvveti hesaplanmalıdır.



Şekil 5.9. Yatay hareketi sağlayan motor miline binen toplam yük

$$Im = \frac{mr^2}{3} \tag{5.34}$$

$$Im = \frac{2200.0{,}4^2}{3}$$

$$Im = \frac{352}{3}$$

$$Im = 117{,}3\ grm^2$$

Bu hesaplamalarda;

Im; Dönme atalet kuvveti (gr.m^2)

m; Kütle (gr)

l; paletin uzunluğu (m) ifade etmektedir.

Yapılan hesaplamalar sonucunda dikey ve yatay hareket için SOYO firmasının ürettiği SY57STH76-2804A (NEMA23) model adım motor tercih edilmiştir. Bu motora ilişkin teknik özellikler aşağıdaki Çizelge 5.3'de görülmektedir.

Çizelge 5.3. Nema23 adım motorunun özellikleri [18]

Özellik	Değer
Bipolar Tutma Torku	1.85 Nm
Tel Sayısı	4
Step/Tur	200
Step Açısı	1.8°
Ağırlık	1 kg
Akım/Faz	2.8 A
Direnç/Faz	1.13 ohm
Rotor Ataleti	480 gr-m^2

Resim 5.20. Nema23 adım motoru

İleri-geri hareket için ise SY42STH47-1684A (NEMA17) model adım motor tercih edilmiştir. Bu motora ilişkin teknik özellikler Çizelge 5.4'de görülmektedir.

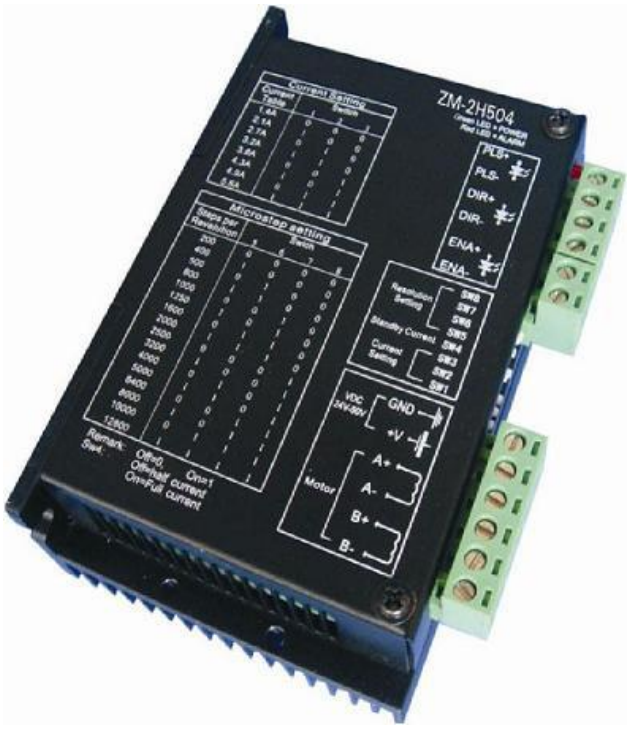
Çizelge 5.4. Nema17 adım motorunun özellikleri [18]

Özellik	Değer
Bipolar Tutma Torku	0.5 Nm
Tel Sayısı	4
Step/Tur	200
Step Açısı	1.8°
Ağırlık	0.50 kg
Akım/Faz	1.68A
Direnç/Faz	1.65 ohm
Rotor Ataleti	68 gcm2

Resim 5.21. Nema17 adım motoru

Nema23 adım motorları sürmek için ZM-2H504 adım motor sürücüsü kullanılmıştır. ZM-2H504 iki faz, 4,6 ve 8 telli step motorlar için üretilmiştir. Yüksek frekanslı giriş sinyallerini kabul edebilecek şekilde donatılmıştır. Akım kararlılığı, çok güçlü parazit önleme kabiliyeti, çok başarılı yüksek frekans performansı,

yüksek başlangıç frekansı, giriş ve çıkış devresi izolasyonu, ayarlanabilir akım, kararlı çalışma, yüksek doğruluk ve düşük gürültülü çalışma gibi üstün özellikleri vardır. 4,2A ve altında, tüm adım motorları rahatlıkla sürebilir. Ürün, gövdeye monte soğutuculu olarak sunulmaktadır.



Resim 5.22. ZM-2H504 adım motor sürücüsü

ZM-2H504 sürücüsünün karakteristik özellikleri şunlardır:

- Maksimum akım 4,2A. Dip switch lerle 8 farklı akım (1A-4,2A) ayarlayabilme. Ayrıca hazırda bekleme akım ayar düğmesi devreye alınırsa, motor çalışmadığı sürece, ayarlanan akımın yarısı gönderilerek, enerji tasarrufu sağlanır ve motorun aşırı ısınması önlenir.
- Entegre aşırım akım koruma devresi.
- En düşük toleranslı ve en yüksek kalitede elektronik bileşenler.
- Ayarlanabilir microstep çözünürlüğü.
- Tüm girişlerde optik izolasyon.
- Aşırı voltaj, faz-faz bağlantısı ve faz-toprak bağlantısı koruması.
- 24-50VDC besleme,
- TTL uyumlu olup mikrodenetleyici veya dijital devreler ile doğrudan sürülebilir.

ZM-2H504 sürücüsünün üzerinde 12 bağlantı terminali ve 8 DIP swich vardır. Bağlantı terminallerinden “GND ve +V“ besleme gerilimin uygulanacağı noktalardır. Daha öncede belirtildiği gibi bu sürücüye, ihtiyaca göre 24 - 50V arasında gerilim uygulanabilmektedir. A+, A-, B+, B- terminallerine ise adım motorun bağlantıları yapılır. Yani adım motora uygulanacak sinyallerin bulunduğu terminallerdir. PLS+ terminali, adım motorun hareketi için sürücüye clock sinyalinin uygulanacağı terminaldir. DIR+ ise adım motorun dönüş yönünün kontrol edildiği terminaldir. ENA+ ise enerjiyi kesmeden motor milini boşa çıkarmak için kullanılır. PLS-, DIR-, ENA- terminalleri ise şaseye bağlanmalıdır.

1-3 arası DIP switch’ler akım ayarı için kullanılmaktadır. Switch’lerin konumuna tekabül eden akımlar Çizelge 5.5’de görülmektedir.

Çizelge 5.5. ZM-2H504 sürücüsü akım ayarları

SW1	SW2	SW3	Akım (A)
ON	ON	ON	1,00 A
OFF	ON	ON	1,46 A
ON	OFF	ON	1,91 A
OFF	OFF	ON	2,37 A
ON	ON	OFF	2,84 A
OFF	ON	OFF	3,31 A
ON	OFF	OFF	3,76 A
OFF	OFF	OFF	4,20 A

4 nolu DIP switch, otomatik akım düşüşü için kullanılır. Otomatik akım düşüşü devre alınırsa, motor durağan halde iken, motorun gereksiz yere ısınmasını ve elektrik harcamasını önlemek amacıyla akım yarıya düşürülür. Motorun harekete başlangıç anında, akım otomatik olarak ayarlanan seviyeye çıkar.

5-8 arası DIP switch’ler mikroadım özelliğini ayarlamak için kullanılır. Normalde 1 tam tur 200 adıma bölünürken, mikroadım özelliği sayesinde çok daha yüksek adımlara bölünebilir. Böylece daha hassas hareket sağlanır. Mikro adım özelliğinin dezavantajı ise motorun üretebileceği tork değerinin düşmesidir. Çizelge 5.6’de

switch'lerin konumuna göre 1 tam turun kaç mikro adıma bölündüğü görülmektedir.

Çizelge 5.6. ZM-2H504 sürücüsünde switch'lerin konumuna göre bir tam turun adımlara bölünmesi

SW5	SW6	SW7	SW8	ADIM/TUR
OFF	ON	ON	ON	400
ON	OFF	ON	ON	800
OFF	OFF	ON	ON	1 600
ON	ON	OFF	ON	3 200
OFF	ON	OFF	ON	6 400
ON	OFF	OFF	ON	12 800
OFF	OFF	OFF	ON	25 600
ON	ON	ON	OFF	1 000
OFF	ON	ON	OFF	2 000
ON	OFF	ON	OFF	4 000
OFF	OFF	ON	OFF	5 000
ON	ON	OFF	OFF	8 000
OFF	ON	OFF	OFF	10 000
ON	OFF	OFF	OFF	20 000
OFF	OFF	OFF	OFF	25 000

Nema17 adım motorunu sürmek için ise ZM-2H042 sürücüsü kullanılmıştır. Bu sürücü ZM-2H504 ile benzer özelliklere sahip olup, 1,7A ve altında tüm adım motorları rahatlıkla sürebilir. Besleme gerilimi 12-24V aralığındadır ve 1,2A yada 1,7A akıma ayarlanabilmektedir.

Resim 5.23. ZM-2H042 adım motor sürücüsü

ZM-2H042 sürücüsünün üzerinde 10 bağlantı terminali ve 4 DIP switch vardır. Bağlantı terminallerinden “GND ve +V“ besleme gerilimin uygulanacağı noktalardır. Daha öncede belirtildiği gibi bu sürücüye, ihtiyaca göre 12 - 24V arasında gerilim uygulanabilmektedir. A+, A-, B+, B- terminallerine ise adım motorun bağlantıları yapılır. Yani adım motora uygulanacak sinyallerin bulunduğu terminallerdir. CP+ terminali, adım motorun hareketi için sürücüye clock sinyalinin uygulanacağı terminaldir. CW+ ise adım motorun dönüş yönünün kontrol edildiği terminaldir. CP- ve CW- terminalleri ise şaseye bağlanmalıdır.

1-3 arası DIP switch’ler mikroadım özelliğini ayarlamak için kullanılır. Normalde 1 tam tur 200 adıma bölünürken, mikroadım özelliği sayesinde çok daha yüksek adımlara bölünebilir. Çizelge 5,7’de switch’lerin konumuna göre 1 tam turun kaç mikro adıma bölündüğü görülmektedir.

Çizelge 5.7. ZM-2H042 sürücüsünde switch'lerin konumuna göre bir tam turun adımlara bölünmesi

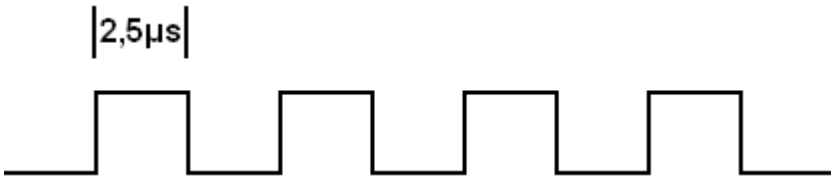
SW1	SW2	SW3	ADIM/TUR
OFF	OFF	OFF	200
OFF	ON	OFF	400
ON	OFF	OFF	800
ON	ON	OFF	1 600
OFF	OFF	ON	3 200
OFF	ON	ON	6 400
ON	OFF	ON	12 800
ON	ON	ON	25 600

4 nolu DIP switch (SW4) ise akım ayarı için kullanılmaktadır. Bu switch, 0 konumunda iken akım değeri 1,2A’e, 1 konumunda iken 1,7A’e ayarlanmaktadır.

5.3.4. Sürücüler ve adım motorlara uygulanan sinyallerin incelenmesi

Adım motorların ve sürücülerinin verilen kontrol sinyallerini takip edebilmeleri ve zamanında tepki gösterebilmeleri girişlerine uygulanacak darbe frekanslarının uygunluğuna bağlıdır.

Kullanılan adım motor sürücülerinin katalog bilgilerinde girişlerine uygulanacak clock sinyallerinin görev süresinin (duty cycle) en az 2,5µs olması gerektiği ve frekans aralığının 0-200KHz olması gerektiği belirtilmiştir.

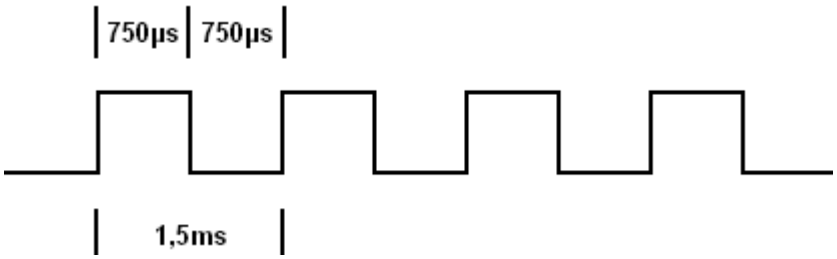


Şekil 5.10. Adım motor sürücüleri için giriş sinyali minimum görev süresi

Mikrodenetleyici yazılımı ile sürücülere uygulanan clock sinyallerinin görev süresi 750µs'dir. Clock sinyali 750µs süresince 1 seviyesinde ve 750µs süresince 0 seviyesinde kalmaktadır. Bu sürelere göre clock sinyalinin periyodu (T) ise 1,5ms olmaktadır. Clock sinyalinin frekansı;

$$f = 1/T = 1/0{,}0015 \cong 666{,}7Hz \tag{5.35}$$

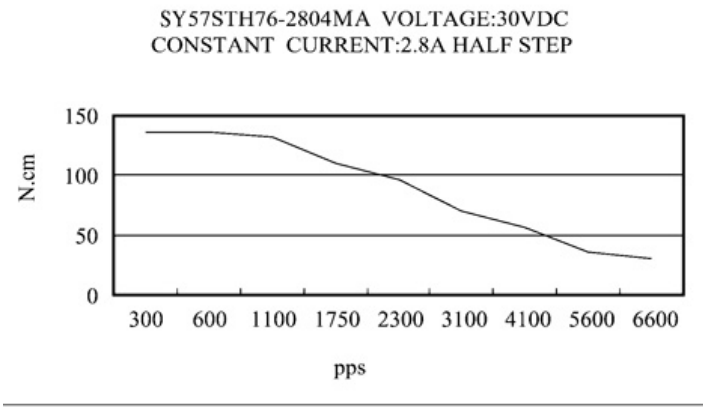
olarak hesaplanır. Bu hesaplamalara göre mikrodenetleyici yazılımı ile sürücülere uygulanan clock sinyalinin hem görev süresinin hem de frekansının uygun olduğu görülmektedir.



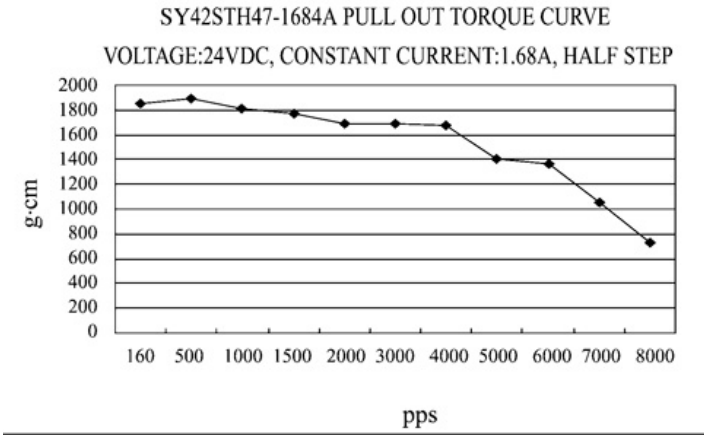
Şekil 5.11. Adım motor sürücülerinin girişlerine uygulanan sinyal

Yapılan çalışmada kullanılan adım motorlardan yatay hareketi sağlayan adım motor bir tam turu 1600 adımda, ileri - geri hareketi sağlayan adım motor ise yine bir tam turunu 1600 adımda tamamlamaktadır. Bu değerlere göre adım motorlara sürücüleri tarafından uygulanan darbe frekansları (pps – pulse per second) 1600Hz'dir. Kullanılan adım motorların tork – frekans eğrileri Şekil 5.12 ve Şekil 5.13'de görülmektedir. Adım motorlara uygulanan sinyalin frekans değerleri arttıkça motorlardan elde edilen tork değeri azalmaktadır.

Şekiller incelendiğinde dikey ve ileri – geri hareketi sağlayan adım motorlara sürücüler tarafından uygun frekans değerlerinde sinyallerin verildiği anlaşılmaktadır.



Şekil 5.12. SY57STH76-2804A (NEMA23) adım motor tork - frekans eğrisi [18]



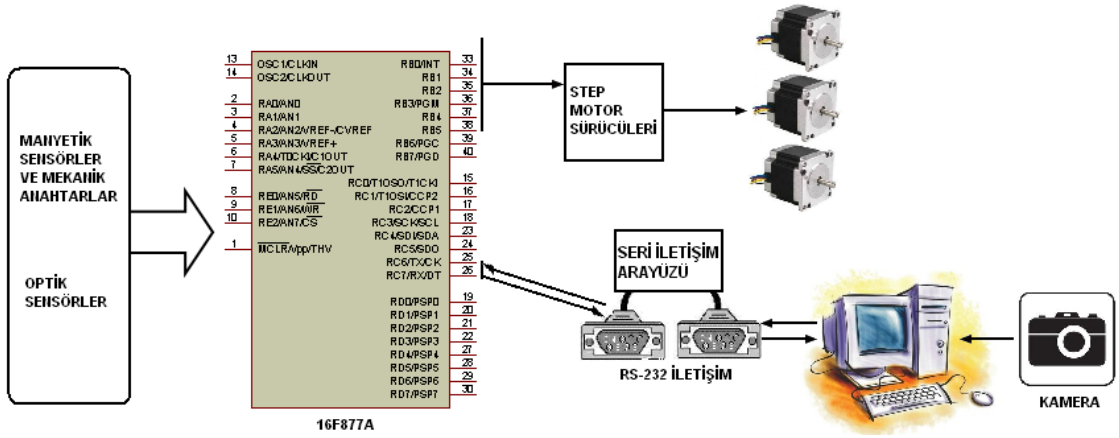
Şekil 5.13. SY42STH47-1684A (NEMA17) adım motor tork - frekans eğrisi [18]

Yatay hareketi sağlayan adım motor ise bir tam turunu 25600 adımda tamamlamaktadır. Bu motora uygulanan sinyalin frekans değerinin, şekilde görülen tork – frekans eğrisine göre uygun olmadığı düşünülebilir. Fakat bu motor yatayda hareket ettiği için Eş. 5.34'de hesaplanan minimum dönme atalet kuvvetini sağlayabilmektedir.

5.4. Elektronik Kontrol Ünitesinin Tasarımı

Otomatik otopark sisteminin elektronik ünitesi üç kısımdan oluşmaktadır. Birincisi sistemin işleyişinden sorumlu, tüm sensörlerin bağlı olduğu, adım motorları

yönlendiren, mikrodenetleyici kontrollü devre; ikincisi mikrodenetleyici ile bilgisayar arasında seri iletişimi sağlayan devre, üçüncüsü ise güç kaynağı (besleme) devresidir. Üç devre de aynı kart üzerine yerleştirilmiştir. Mikrodenetleyicili devre ile seri iletişim devresi birbiri ile iletişim kurabilmektedir. Otomatik otopark sisteminin görsel blok şeması Şekil 5.14'de görülmektedir.



Şekil 5.14. Otomatik otopark sisteminin görsel blok şeması

Mikrodenetleyici sensörlerden aldığı bilgileri yorumlayarak adım motorları kontrol etmekte ve bilgisayar yazılımı ile iletişim kurmaktadır. Kullanılan sensörler; manyetik röleler, kızılötesi yakınlık sensörleri, mekanik anahtarlardır. Seri iletişim için MAX232 entegresi kullanılmıştır. Mikrodenetleyici programı CCS C ile yazılmış, bilgisayar yazılımı ise C# ile geliştirilmiştir. Kamera olarak basit bir webcam kulanılmıştır.

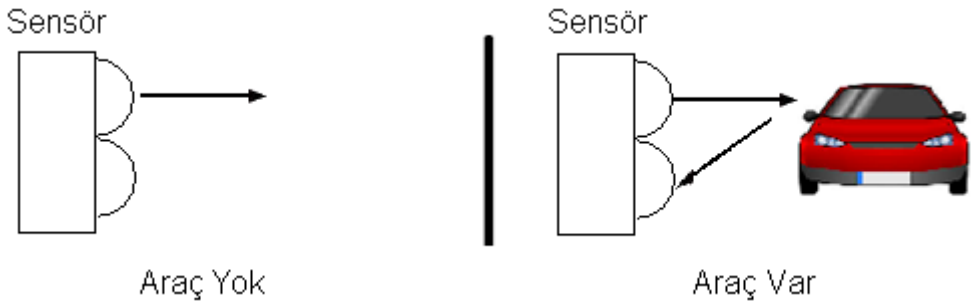
5.4.1. Mikrodenetleyicili kontrol biriminin tasarımı

Mikrodenetleyici olarak Microchip firmasının ürettiği 16F877A entegresi kullanılmıştır. Bu mikrodenetleyicinin seçilmesinin başlıca nedenleri çok sayıda giriş/çıkış portuna sahip olması, seri iletişim biriminin olması, program belleğinin ideal olması, oldukça çok sayıda uygulama örneği bulunmasıdır. 16F877A ile ilgili geniş bilgi 4. bölümde verilmiştir.

Mikrodenetleyicinin 33 iletişim portunun 26'sı aktif olarak kullanılmıştır. Aktif olarak kullanılan portların 14 tanesi kızılötesi yakınlık sensörlerine, 2 tanesi mekanik anahtarlara, 2 tanesi manyetik anahtarlara bağlanmış; 6 tanesi ise step motor

sürücü birimlerine bağlanarak step motorların kontrolü amaçlanmıştır. 2 tanesi de bilgisayar yazılımı ile seri iletişim kurmak için kullanılmaktadır. A, B, C ve D portları kullanılmıştır.

Kart dizaynının anlatılmasından önce sensörlerin ne amaçla kullanıldığından bahsedilecektir. Optik sensör olarak kızılötesi yakınlık sensörü kullanılmıştır. Bu sensör araçların park edileceği bölmelere ve araç kabul bölmesine yerleştirilmiştir. Her bir sensör kullanıldığı bölmenin boş ya da dolu olduğu bilgisini mikrodenetleyiciye bildirmektedir.



Şekil 5.15. Yakınlık sensörünün aracı algılaması

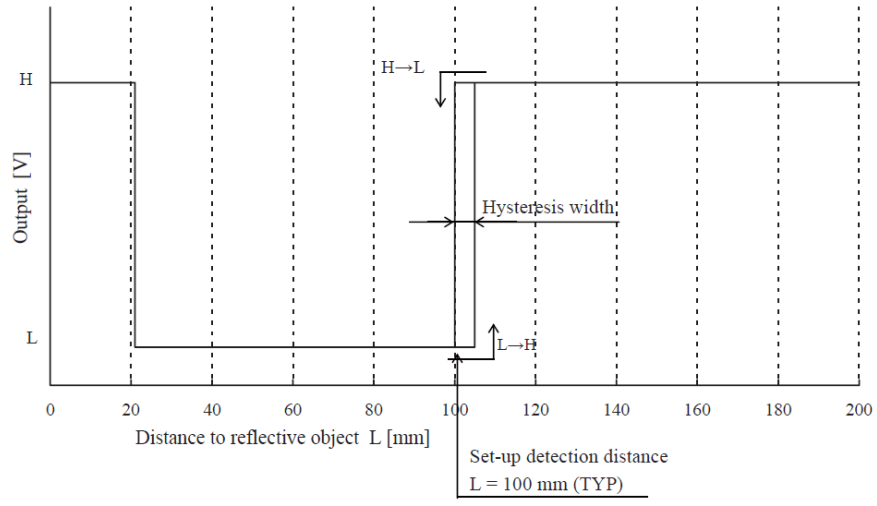
Projede sensör olarak Sharp GP2Y0D810Z0F kullanılmıştır. Sharp GP2Y0D810Z0F; algılama menzili olan 2-10cm mesafede bulunan objeleri algılayabilen dijital bir kızılötesi yakınlık sensördür. Küçük boyutları, hızlı tepki süresi, düşük akım gereksinimi ve çok yakındaki objeleri algılayabilmesi sayesinde özellikle temassız, yakın mesafe nesne algılamaları için oldukça kullanışlıdır.
Tepki süresi analog kızılötesi sensörlerden yüksek olması dolaysıyla, düşük mesafeler için iyi bir alternatiftir. Ancak bu sensör nesnelerin bulunduğu konumu belirlemez. Sadece menzil içerisinde var olup olmadığını tespit edebilir. Algılama menzili içinde bir nesne yok ise çıkışı 1, eğer nesne var ise çıkışı 0'dır. Resim 5.24'de Sharp GP2Y0D810Z0F sensörü görülmektedir.

Resim 5.24. Sharp GP2Y0D810Z0F sensörü

Sensör kızılötesi LED (IRLED), pozisyon dedektörü (position sensitive dedector - PSD) ve sinyal işleme devresinin kombinasyonundan oluşmuştur. Ortam sıcaklığı ve çalışma süresi sensörün kararlı çalışmasını etkilememektedir. Aşağıdaki tabloda sensörün karakteristik özellikleri görülmektedir [36].

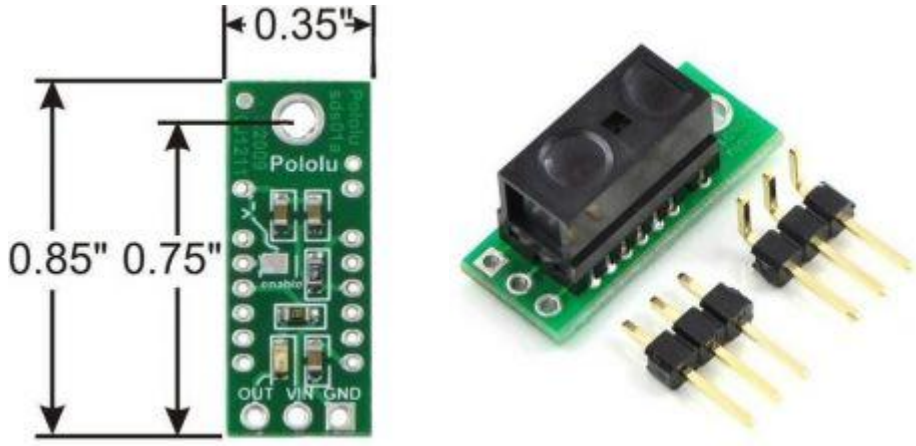
- Dijital çıkış,
- 2-10cm algılama mesafesi,
- Boyutları : 13.6×7×7.95 mm,
- Tipik akım değeri 5mA,
- Besleme voltaj aralığı 2,7 - 6,2V,
- Günışığı toleransı.
- Tepki süresi 2,56ms (3,77ms max)
- Örnekleme hızı 390Hz
- Ağırlığı 1,3gr (pinler hariç)

Sensörün dataseet bilgilerine göre kullanım esnasında doğrudan güneş ışığı veya diğer ışık kaynaklarına maruz bırakılması hatalı algılama sonuçları doğurabilmektedir. Ayrıca cisimlerin yüzeyi parlak ise (ayna vb.) yine algılama sonuçları hatalı olabilmektedir. Lehimleme işlemi 260^{o}C'nin altında olmalı ve 5sn'den daha az olmalıdır. En fazla 2 kez lehim yapılmalıdır. Şekil 5.16'de cismin algılanması esnasında çıkış sinyalindeki değişim grafiği görülmektedir [36].



Şekil 5.16. Cisim algılaması durumunda sensörün çıkış sinyalindeki değişim [36]

Sensörün kullanımını kolaylaştırmak için hazır taşıyıcı kart kullanılmıştır. Bu kartlar “breakout card” yada “carrier card” olarak da isimlendirilmektedir. Taşıcı kart üzerinde besleme için iki ve çıkış için bir olmak üzere toplam üç adet pin çıkarılmıştır. Resim 5.25’de boş taşıyıcı kart ve sensörün monte edilmiş hali görülmektedir.



Resim 5.25. Sensörün taşıyıcı kartı ve boyutları

Kullanılan bir diğer sensör ise manyetik röle ve mıknatıs çiftidir. Reed röle yada dil rölesi olarak bilinen bu sensör; bir cam tüp içinde iki ucu bağlantı için dışarı çıkarılmış normalde açık bir anahtardır. Mıknatıs bu anahtara etkin mesafede yaklaştığında kontaklar manyetik alandan etkilenerek kapanmaktadır.

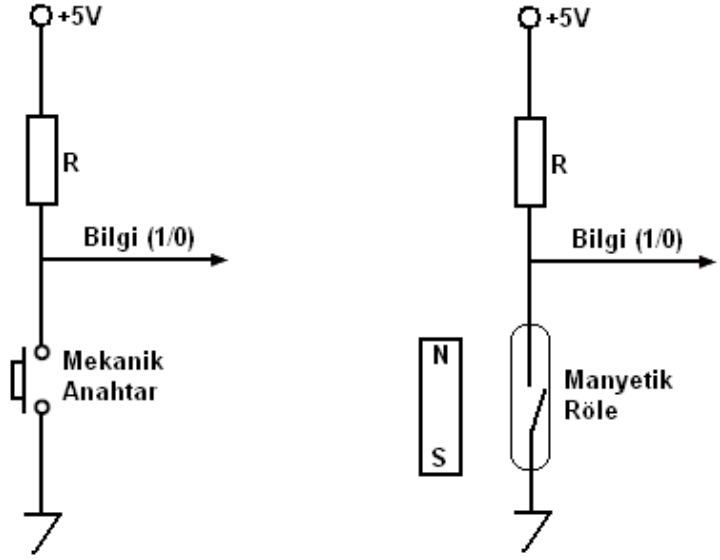
Resim 5.26. Reed röle

Manyetik röleler uygulamada iki yerde kullanılmaktadır. Y ekseninde ileri-geri hareket eden paletin hareket sınırları iki adet manyetik röle ile belirlenmektedir.

Başlangıç pozisyonu ile sisteme ilk enerji verildiğinde paletin araç kabul bölmesinin önünde uygun konuma gelmesini sağlamaktadır. İki adet mekanik anahtar bu amaçla kullanılmaktadır. Kullanılan mekanik anahtarlara piyasada sınır anahtarı da denilmektedir. Başlangıç pozisyonuna geçmek için önce açık durumda ise alttaki mekanik anahtarın kapanması, sonra kulenin sağına (karşıdan bakınca

bize göre sol) yerleştirilen ve açık olan ikinci mekanik anahtarın açılıp kapanması ve son paletin belirlenen adım sayısı kadar (belirlenen açıda) sağa doğru hareket ederek otoparkın kabul bölmesinin önüne gelmesi gerekmektedir.

Sistem çalışmak için enerji aldığında önce alttaki mekanik anahtarın kapalı olup olmadığı kontrol edilir. Açık ise kapanana kadar mekanizma aşağı hareket eder. Mikrodenetleyiciye anahtarın kapalı olduğu bilgisi ulaştıktan sonra bu anahtarın görevi tamamlanır. Sonra palet sistemi yatayda sağa doğru dönmeye başlar. Bu hareket kulenin sağına yerleştirilen mekanik anahtara palet çarpıp kapanıncaya kadar devam eder. Anahtar kapanıp mikrodenetleyiciye bilgi ulaştığında palet bu sefer yatayda sola doğru dönmeye başlar. İkinci anahtar da tekrar açık konuma geçer ve görevi tamamlanır. Sağa doğru dönme hareketi paletin kabul bölmesinin önüne gelmesi ile son bulur ve sistem başlangıç pozisyonuna geçmiş olur. Park için araç kabul etmeye hazırdır.



Şekil 5.17. Mekanik anahtar ve reed rölenin mikrodenetleyiciye bilgi sağlaması

Diğer iki manyetik röle, yukarıda bahsedildiği gibi, paletin ileri/geri hareket sınırlarını belirler. Palet aracı almak için altına doğru yeteri kadar uzadığında (ileri hareket), paletin soluna yerleştirilmiş olan mıknatıs ikinci manyetik röleyi kapatarak mikrodenetleyicinin paleti durdurmasını sağlar. Palet arabayı alıp yeteri kadar geri geldiğinde ise paletin sağındaki bir başka mıknatıs diğer manyetik röleyi kapatarak

yine paletin durması sağlanır. Böylece paletin ileri/geri hareket aralığı bu manyetik röleler tarafından belirlenir.

Redundant (yedek, ikinci) sensörler

Kullanıldığı yere ve fonksiyonuna göre sensörün görevini yerine getirmemesi ciddi bir olumsuzluğa neden olmayabileceği gibi çok önemli kaza ve kayıplara de neden olabilir. Kaza ve kayıpların olabileceği durumlarda sensörün bozulma veya görevini yapamama ihtimaline karşı aynı ya da benzer şekilde görevi üstlenen başka bir sensör emniyet amacı ile kullanılır. Bu sensörlere “redundant sensör”ler denir. Redundant kelimesi; gereksiz, gereğinden fazla, ihtiyaç fazlası anlamlarına gelmektedir. Bir anlamda yedek sensör de denilebilir. Örneğin hareketli bir sistemde hareket sınırı bir limit anahtarı ile belileniyorsa, bu anahtarın bozulması durumunda hareket durdurulamayacaktır. Bu durum hareketli mekanizmanın çevreye, insanlara veya kendine zarar vermesine neden olabilir. Kullanılan limit anahtarının görevini yerine getirememesi halinde onun görevini devralacak ikinci bir sensör kullanılabilir. Bu sensöre redundant sensör adı verilir.

Redundant sensörler özellikle havacılık, robotik vb. uygulamalarda sistem güvenilirliğini artırmak için kullanılır. Redundant sensör kullanımının bariz avantajı basit bir yöntem ile tüm sistemin güvenilirliğini artırmasıdır. Redundant sensörler, diğer sensörlerden kaynaklanabilecek anormal davranışlardan sistemin daha az etkilenmesini veya hiç etkilenmemesini sağlarlar. Bir başka deyişle redundant sensörler diğer sensörlerin kişisel hatalarına karşı sistemi korurlar [37, 38].

Algılama robot sistemlerinde en önemli fonksiyonlardan biridir. Redundat ve çoklu algılama, robot performansını artırmak ve robot zekasını geliştirmek için daha fazla bilgi sağlar. Algılama elemanları, iş gören elemanların (working objects) ve çevresel şartların özelliklerini ve karakteristiklerini algılayarak robotların görevlerini yerine getirmelerini ve beklenmeyen durumlardan sakınmalarını sağlarlar [39].

Redundant sensör olarak kullanılan sensörler asıl sensörler ile özdeş olabilecekleri gibi aynı amaca hizmet eden farklı sensörler de olabilirler. Aynı yerde konumlanmış olabilecekleri gibi farklı yerlerde de konumlanabilirler [39].

Redundant sensörlerin etkili kullanımı için, sensörlerden elde edilen farklı seviyelerde ve farklı formatlarda bilgilerin birleştirilmesi daha iyi sonuç verir [39].

Redundant algılamanın; güvenilirliği artırması, verimi artırması,, self kalibrasyon ve sensörlerdeki yıpranmadan (eskimeden) oluşabilecek hataları tolare etmesi gibi avantajları vardır [39].

Bu tezde yapılan çalışma prototip niteliğinde olduğu için herhangi bir yerde redundant sensör kullanılmamıştır. Fakat sistemin güvenilirliğini artırmak birkaç yerde redundant sensör kullanılabilir. Örneğin paletin ileri – geri hareket sınırlarını belirleyen manyetik anahtarların görevini yerine getirememesi durumunda araçların zarar görmemesi için paralel başka manyetik anahtarlar veya paletin hareketini durduracak limit anahtarları kullanılabilir.

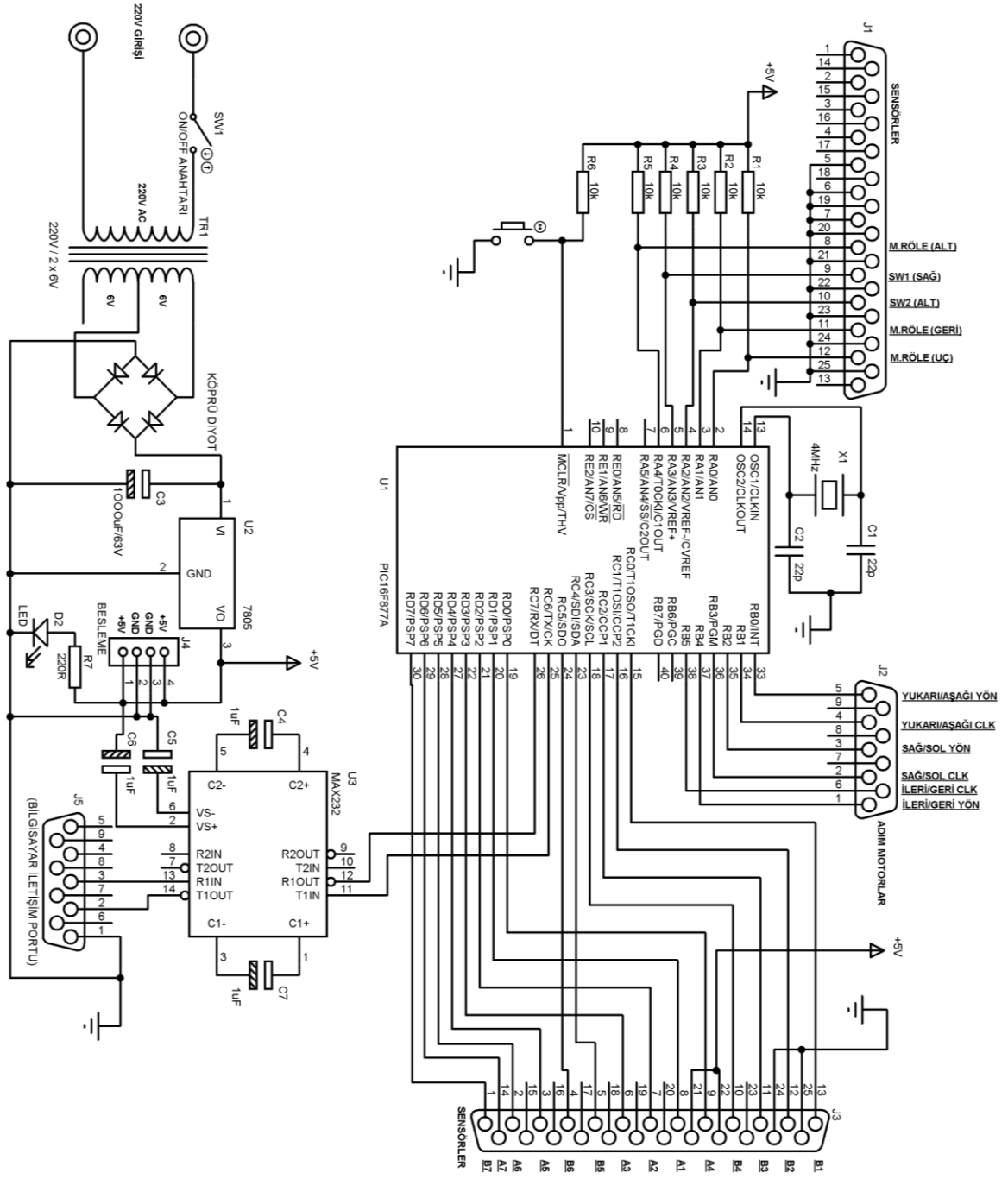
Elektronik devre şemasının çizimi

Elektronik baskı devre kartı dizaynı için Proteus programı kullanılmıştır. Labcenter tarafından üretilen Proteus görsel olarak elektronik devrelerin simülasyonunu yapabilen bir devre çizim, simülasyon ve PCB çizim programıdır. İki temel bileşenden oluşur.

- *ISIS (Intelligent Schematic Input System):* Elektronik Devre çizim, simülasyon ve animasyon programı,
- *ARES (Advanced Routing & Editing Software):* Baskı devre çizim programı.

ISIS'in klasik yazılımlardan en önemli farkı mikroişlemci ve mikrodenetleyicilere yüklenen kaynak kodlarını simüle edebilmesidir. Gün geçtikçe genişleyen bir model kütüphanesine sahiptir.

Elektronik devrelerin şemalarının çizildiği ISIS programında hem devre şeması çizilmiş hem de olabildiğince benzetim (simülasyon) yapılmıştır. Daha sonra ARES programı ile baskı devre çizimi gerçekleştirilmiştir. Şekil 5.18'de devre şeması görülmektedir.



Şekil 5.18. Elektronik kontrol ünitesinin devre şeması

Devrede kullanılan R1…R5dirençleri 10KΩ değerindedir ve mikrodenetleyiciye 1/0 bilgilerini verebilmek amacı ile sensörlere bağlı gerilim bölücüler olarak kullanılmışlardır. 16F877A'nın 1 nolu pini reset (master clear) girişi olup B1 butonu ve 10KΩ'luk direnç (R6) kombinasyonu ile gerektiğinde devre resetlenebilmektedir. Reset girişine 10KΩ'luk direnç üzerinden +5V uygulanarak reset girişi pasif tutulmaktadır. B1 butonuna basıldığında devre resetlenir. Mikrodenetleyicinin yüklenen programı işleyebilmesi için gerek duyduğu clock sinyali 4MHz kristal (X1) ve 2 adet 33pF (C1 ve C2) kondansatör kombinasyonu ile sağlanmaktadır.

J4 konnektörü devrenin beslenme uçlarının dışarı çıkarıldığı terminallerdir. 1 ve 4 nolu uçlarında +5V, 2 veya 3 nolu uçlarında ise şase (GND) potansiyeli mevcuttur. Bu çıkışlar normalde gerekli değildir ve devrenin çalışması ile bir ilgisi yoktur. Gerektiğinde kullanıma (şaseyi birleştirme, ölçüm yapma vb.) hazır olması amaçlanmıştır. Gerekmezse kullanılmayacaktır.

Adım motorlarla ve sensörlerle bağlantıyı sağlamak için J1, J2 ve J3 konnektörleri kullanılmıştır. J2, seri port konektörü olarak da bilinen dişi DB9 konnektörüdür. Adım motorlar ile bağlantı J2 konnektörü ile sağlanmaktadır. Mikrodenetleyicinin RB0, RB1, RB2, RB3, RB4, RB5 portları adım motorları kontrol eden sinyalleri üretir. Bu sinyaller dönüş yönünü belirleyen ve hareketi sağlayan sinyallerdir. Yön için 0 yada 1 bilgisi sürekli olarak uygulanır. Dönme hareketi için ise sürekli clock sinyali verilmelidir. Clock sinyalinin frekansı dönüş hızını etkiler. Adım motorlar bir sürücü vasıtası ile kontrol edildiğinden mikrodenetleyicinin ürettiği bu sinyaller sürücü birimine uygulanmaktadır. Çizelge 5.8'de portların ürettiği sinyalleri hangi adım motorun ne amaçla kullandığı görülmektedir.

Çizelge 5.8. J2 konnektöründen adım motorlara sağlanan sinyaller

J2 Pin No	Port	Üretilen sinyalin görevi
1	RB4	İleri/geri hareketi sağlayan adım motor için yön bilgisi
2	RB3	Yayay hareketi sağlayan adım motor için clock sinyali
3	RB2	Yatay hareketi sağlayan adım motor için yön bilgisi
4	RB1	Dikey hareketi sağlayan adım motor için clock sinyali
5	RB0	Dikey hareketi sağlayan adım motor için yön bilgisi
6	RB5	İleri/geri hareketi sağlayan adım motor için clock sinyali

J1 ve J3 dişi DB25 konnektörüdür. Sensörler ile bağlantı J1 ve J3 konnektörleri vasıtası sağlanmaktadır. İki konektöre bağlanan toplam sensör sayısı 18'dir. Tek bir konektöre 25 bağlantı yapılabildiğine göre 2 konnektör kullanmak yersiz gibi gelebilir. Tek konnektör kullanıldığında baskı devre tasarımı oldukça zorlaşmakta ve çok sayıda atlama yapmak gerekmektedir. Ayrıca sensörlere gerekli olan beslemeler de bu konektörler vasıtası ile yapılmaktadır. J1 konnektörünün 21, 22, 23, 24, 25 nolu pinleri manyetik röleler ve mekanik anahtarlar için şase bağlantısını sağlamaktadırlar. Bu sensörlerin bir uçları ilgili portlara bağlı iken diğer uçları şaseye bağlıdır. J3 konnektörünün 21, 22, 24, 25 nolu pinleri ise kızılötesi yakınlık sensörlerini için besleme gerilimini sağlamaktadırlar. Çizelge 5.9'de J1 ve J3 konnektörlerinin hangi pinlerinin hangi sensörlere bağlı olduğu görülmektedir.

Çizelge 5.9. J1 ve J3 konnektörleri sensör bağlantıları

Konnektör	Pin No	Port	Bağlı Olduğu Sensör
J1 KONNEKTÖRÜ	9	RA3	SW1 (Sağ)
	10	RA2	SW2 (Alt)
	11	RA1	M.Röle-1 (Geri)
	12	RA0	M.Röle-2 (Uç)
	22		SW1
	23		SW2
	24		M.Röle-1
	25		M.Röle-2
	5	GND	GND'yi başka bir noktaya (PIC ve OSC devresi) taşımak için kullanılmışlardır.
	6		
	7		
	19		
	20		
J3 KONNEKTÖRÜ	1	RD7	B7 Yakınlık Sensörü
	2	RD5	A6 Yakınlık Sensörü
	3	RD4	A5 Yakınlık Sensörü
	4	RC5	B6 Yakınlık Sensörü
	5	RC4	B5 Yakınlık Sensörü
	6	RD3	A3 Yakınlık Sensörü
	7	RD2	A2 Yakınlık Sensörü
	8	RD1	A1 Yakınlık Sensörü
	9	RD0	A4 Yakınlık Sensörü
	10	RC3	B4 Yakınlık Sensörü
	11	RC2	B3 Yakınlık Sensörü
	12	RC1	B2 Yakınlık Sensörü
	13	RC0	B1 Yakınlık Sensörü
	14	RD6	A7 Yakınlık Sensörü
	21	+5V	Kızılötesi Yakınlık Sensörlerinin Beslemesi
	22		
	24	GND	
	25		

5.4.2. Seri iletişim biriminin tasarımı

İkinci elektronik devre ise hem mikrodenetleyicili devre ile hem de bilgisayar yazılımı ile iletişimi sağlamaktadır. İlerleyen bölümlerde de anlatılacağı üzere otoparka gelen araçların bir webcam ile fotoğrafları çekilip plaka tanımlaması yapılmakta ve hangi bölmeye yerleştirildiğinin kaydı tutulmaktadır. Ayrıca giriş çıkış zamanına göre süre ve ücret hesaplaması yapılmaktadır. Tüm bu işlemler bilgisayar üzerindeki yazılım vasıtası ile yapılmaktadır. Seri iletişimi sağlayan elektronik devre, anlatılan işlemlerin yapılması ve aracın yerleştirip teslim edilmesi aşamalarının hatasız olarak yapılabilmesi için mikrodenetleyicili devre ile bilgisayar yazılımı arasında bir arayüz oluşturarak haberleşmelerini sağlamaktadır.

PC tabanlı bilgisayarlar ile çevre birimleri arasında haberleşmeyi sağlayabilmek için kullanabilen bazı iletişim portları mevcuttur. Birçok PC tabanlı bilgisayarın kasasında bulunan, LPT, COM, USB portlarını kullanılarak çevre birimleriyle rahatlıkla haberleşilebilir. Bu portların hızları, adresleri, pin sayıları, teknik özellikleri ve bağlantı noktaları birbirinden farklıdır. Aşağıda bu portlar hakkında özet bilgiler yer almaktadır.

LPT portu

Paralel yazıcı portu olarak bilinen LPT portu, 25 pinden oluşmaktadır. IBM tabanlı bilgisayarlardaki paralel port, 3 adet 8 bitlik portla işlemcinin I/O'suna (giriş/çıkış birimi) erişebilen, 12 sayısal çıkışa ve 5 sayısal girişe sahiptir [40].

Bilgisayarlarda en fazla üç paralel port bulunur. Hangi porta kurulduğuna bağlı olarak, paralel portun adresi 278h, 378h veya 3BCH olabilir. Tek paralel portu olan bilgisayarlarda port adresi genellikle 378h adresinde bulunmaktadır. Paralel port, paralel port arabirimine sahip yazıcıları bilgisayara bağlamak için tasarlanmışlarsa da, giriş-çıkış karakteristiği uyumlu herhangi bir cihaz veya uygulamaya yönelik olarak genel amaçlı giriş/çıkış portu olarak da kullanılabilmektedirler [40].

COM portu

Bilinen bilgisayarların seri iletişimleri UART olarak adlandırılan bir birim üzerinden gerçekleştirilmektedir. Seri olarak gönderilecek bilgi UART'ın gönderme kaydedicisine koyulur. UART birimi veriyi yapılan ayarlara göre bit bit gönderir. Tersi olarak UART, karşı taraftan gelen bilgiyi alma kaydedicisine kaydeder ve okunana kadar orada bulundurur. UART içerisinde birçok kaydedici mevcuttur. Bu kaydedicilere port adresleri yardımıyla erişilir. Örneğin standart bir bilgisayarda COM1 için adres 0X3F8, COM2 için 0X2F8'dir [41].

USB portu

Hızlı ve güvenilir veri iletimi, esneklik, düşük maliyet ve güç tasarrufu USB'nin kullanıcılarıa sunduğu avantajlardandır. USB (universal serial bus) cihazlarda kullanıcıların ayar yapmasına gerek kalmaz. Örnek vermek gerekirse, port adresleri kesme isteği kanalları (IRQ) ile uğraşmaya gerek yoktur. USB kullanımıyla, cihazların ihtiyaç duymadığı IRQ'lar boşaltılır. USB arabirimine bir IRQ hattı ile bir dizi port adresi tahsis edilir. Çevre birimleri gibi ilave besleme kaynağına ihtiyaç duymazlar. USB olmayan çevre birimleri ise özel port adresleri, çoğu kez bir IRQ hattı ve kimi zaman da genişleme yuvasına ihityaç duyarlar [42].

USB arabirimi, bilgisayardan elde ettiği +5V gerilim ve toprak hattı sayesinde çoğu zaman harici güç kaynağına ihtiyaç duymaz, 500 mA akım sağlayabilir. USB üç hızı destekler. Yüksek hız (saniyede 480 Megabit), tam hız (saniyede 12 Megabit) ve düşük hız (saniyede 1.5 Megabit) [42].

Proje açısından iletişim portlarının karşılaştırması

Proje açısından bilgisayar iletişim portlarının çeşitli özellikleri karşılaştırıldığında, elektronik kontrol devresi ile bilgisayar iletişimi için COM portun kullanılmasının uygun olacağı ortaya çıkmaktadır. USB portun çok yüksek hızlı iletişimlerde kullanılmasının daha uygun olacağı düşünülmüştür. Çizelge 5.10'da iletişim portlarının bir karşılaştırması görülmektedir.

Çizelge 5.10. İletişim portları karşılaştırma Çizelgesi [40 - 42]

Arabirim	Format	Cihaz Sayısı	Uzunluk (feet)	Hız (bit/saniye)
USB	Asenkron seri	127	16 (5 hub ile 96)	1,5M 12M 480M
RS-232 (COM)	Asenkron Seri	2	50-100	20K (Donanımla 115K'ya kadar)
LPT	Paralel	2 (papatya zinciri ile 8)	10-30	8M

RS-232 İletişiminin Temel Kavramları

RS-232, bilgisayarın harici cihazlarla haberleşmede kullanılandığı iletişim standartlarından biridir. RS-232 temel olarak bir seri iletişim birimidir. Seri iletişim biriminde bilgiler bir hat üzerinden bit bit yollanır. Paralel iletişime göre en önemli avantajı bağlantı kolaylığıdır. Bilgisayardan cihaza veri göndermek için bir hat, cihazdan gelen verileri almak üzere bir hat ve bir toprak hattı olmak üzere toplam üç hat kullanılarak iletişim sağlanır. Bu standardın önemli dezavantajı ise iletişim hızı arttıkça bilgi kaybını önlemek için kablo uzunluğunun kısalması gerekliliğidir. Standard RS-232 iletişimi 19200 baud haberleşme hızında en fazla 20m kablo uzunluğuna izin vermektedir. Seri iletişim standartlarından RS-422, RS-449 daha yüksek iletişim hızlarında çok daha uzun kablolamaya imkan sağlamaktadır. Örneğin RS 422, 1600 m uzunluğunda bir kablo üzerinden bir megabit/saniye haberleşme hızını desteklemektedir [43].

Seri veri iletişimi

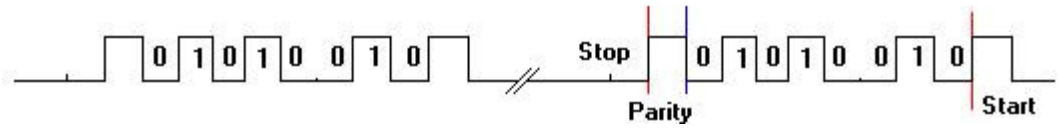
Seri haberleşmede 8 ya da daha farklı sayıda veriler iki hat üzerinden taşınırlar. Paralel haberleşmeye göre daha uzun mesafelere veriler iletilebilir. Paralele göre dezavantajı ise veri iletim hızının yavaş olmasıdır. Seri haberleşmede önemli terimler aşağıda açıklanmıştır [44].

Baud Rate: Veri iletim hızı olup bir saniyede iletilen veri adedine denir. Standart olarak veri hızları 300, 600, 1200, 2400, 4800, 9600, 19200, ... seklindedir.

Start Biti: Asenkron veri iletişiminde bir veri gönderildikten sonra yeni bir veri herhangi bir zamanda gelebilir. İşte bu yeni verinin başlangıcı start biti ile bildirilir.

Stop Biti: Gönderilen verinin bittiğini belirten bittir. Bu biti alan alıcı yeni bir veri için start bitini beklemeye başlar. Stop biti iletişimin özelliğine göre 1 yada 2 bit uzunluğunda olabilir.

Eşlik Biti (Parity Bit): Hata denetimine yonelik kullanılan bitdir. Start ve 8 bitlik bir veri iletildikten sonra stop biti gönderilmeden önce parity gönderilir. Tek veya çift (EVEN, ODD) parity kullanılmasına göre partiy biti 1 veya 0 olabilir. Alıcı bu biti kontrol ederek alınan verinin doğru olup olmadığını belirler. Parity biti, 1 bit uzunluğundadır.



Şekil 5.19. Seri veri iletişimi

Burada dikkat edilmesi gereken Start, stop ve veri bitlerinin süresidir. Bu da baud rate'i verir. Örnek olarak parity bitinin kullanılmadığı; bir start, bir stop ile birlikte 8 bitlik bir bilgi (toplam 10 bit) 2400 baud hızında gönderilirse veri paketinin iletim süresi ve bir bitin iletim süresi Eş. 5.36 ve Eş. 5.37'deki gibi hesaplanır.

Veri paketi iletim süresi $= \frac{1}{2400} \cong 417\mu S$ (5.36)

Bir bit iletim süresi $= \frac{417}{10} = 41{,}7\mu S$ (5.37)

Seri veri transferinin paralele göre avantajları şunlardır [43, 44]:

- Seri kablolar paralel kablolardan daha uzun olabilir. Seri port çıkışında '1' biti -3 ile -25 volt, '0' biti ise +3 ile +25 volt aralığında temsil edilir. Paralel port kullanımında ise '0' 0 - 2,5 volt, '1' ise 3 - 5 volt aralığında temsil edilir. Bu nedenle, paralel porttaki maksimum sinyal salınımı 5 volt iken seri portta bu 50 volta kadar çıkabilmektedir. Bu nedenle seri portta kablodaki gerilim kaybı paralel porttaki kadar fazla problem teşkil etmez.

- Paralel iletişimmdeki gibi fazla kabloya ihtiyaç olmaz. Eğer bağlanacak cihaz bilgisayardan uzak bir mesafede ise 3 kablo kullanmak, paralel iletişimdeki gibi 19 veya 25 kablo kullanmaktan daha ekonomiktir.
- Infra Red (Kızıl Ötesi) iletişimde 8 bitlik bir verinin aynı anda bir oda içinde iletilmesinin imkânsız olduğu açıktır. Bundan dolayı kızılötesi haberleşmede seri iletişim kullanılır.
- Mikrodenetleyici kullanımı da günümüzde oldukça yaygınlaşmıştır. Bunların birçoğu dış dünya ile iletişim kurmak için SCI (Serial Communication Interface - Seri İletişim Arabirimi) ünitelerine sahiptir. Seri iletişim mikrodenetleyicilerde kullanılacak pin sayısını azaltır. 8 bitlik paralel iletişimde 8 pin kullanılması gerekirken seri iletimde yalnızca üç pin ortak olarak kullanılır.

Veri iletim hatları

Hatlar bilgi iletişimine olanak sağlayan ortamlardır. Bu ortamlarda veri iletilirken band genişliğinin dar olması ve hattın kapasitesinin sınırlı olması gibi sorunlar ortaya çıkabilir. Haberleşme hatları Simplex, Half-dublex ve Full-dublex olarak üç gruba ayrılabilir.

Simplex Hatlar: Bir verici ve bir alıcının olduğu sistemlerdir. Veri iletişimi tek yönlüdür. Verici bilgiyi gönderir ve alıcı da sadece gönderilen bilgiyi alır. Vericiye herhangi bir bilgi gönderilemez. Örneğin; TV sistemlerinde verici tek yönlü olarak veriyi gönderir ve alıcı da alır.

Half Dublex Hatlar: İletişim iki yönlüdür. Veri iletişim hattından bilgi gönderimi sırayla yapılır. Bir taraf veriyi gönderirken diğer taraf sadece dinleme yapabilir. Sonra roller değişir. Telsiz konuşmaları half dublex iletişime örnek olarak verilebilir.

Full Dublex Hatlar: Yine çift yönlü iletişim vardır. Hattan aynı anda veri gönderimi ve alımı yapılabilir. Telefon iletişimi buna en iyi örnektir.

Senkron - asenkron veri iletimi

Cihazların haberleşmesi için veri iletiminden önce birbirlerini uyarmaları gerekir. Veri gönderilmeden önce alıcı uyarılmazsa, alıcı gelen bit serisi için gerekli zamanı ayıramaz. Bu nedenle zamanlama sinyalleri çok önemlidir. Gönderilen veri dizisi alıcı tarafından yanlış zamanda incelenirse gelen bilgi yanlış anlaşılabilir. Bilginin uygun zamanda alınmasına senkronizasyon (eşzamanlı) adı verilir. Senkronizasyonu sağlamak için saat (clock) sinyalleri kullanılır [43].

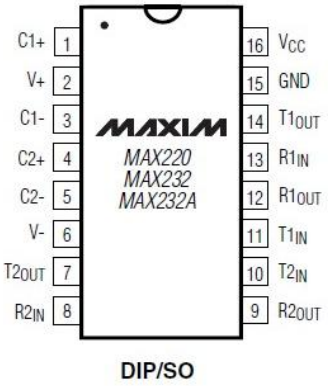
Senkron veri iletimi: Senkron veri iletiminde veri bloğunun başındaki ve sonundaki bildiri karakterleri dışında alıcı ve verici arasında clock sinyali taşıyan ayrı bir hat kullanılır. Bu hattan uygulanan clock sinyali alıcı ve vericinin senkron çalışmasını sağlar. Senkron iletişimin sonunda "iletim sonu" sinyali gönderilir.

Asenkron veri iletimi: Asenkron iletime başla-bitir iletimi adı da verilmektedir. Asenkron iletimde veri ile clock sinyali kullanılmaz. Senkronizasyon için start ve stop bitleri kullanılır. Çünkü veri herhangi bir zamanda gönderilebilir. Start ve stop bitleri veri bloğunun başında ve sonunda kullanılır.

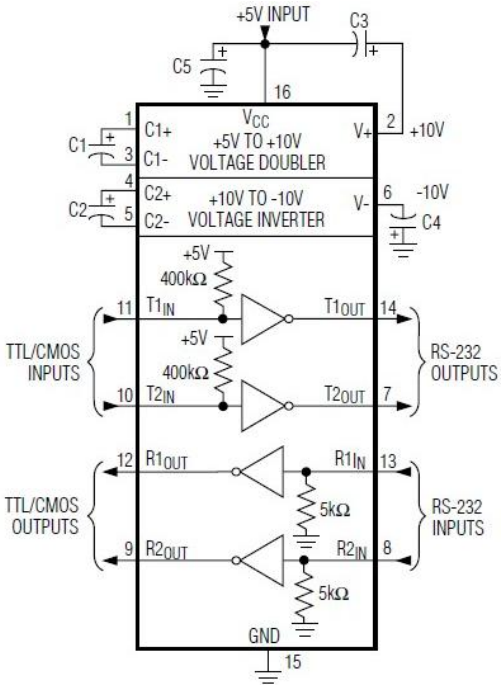
MAX 232 entegresi

Seri iletişim sağlamak için MAX232 entegresi kullanılmıştır. Çünkü RS232 standartları, TTL entegrelerinden daha önce belirlendiğinden gerilim seviyeleri bakımından TTL uyumlu değillerdir. RS232'de lojik-0, +3V ile +25V arasında, lojik-1 ise -3V ile -25V arasında tanımlanmaktadır. Bu seviyeler TTL ile uyumlu değildir. Mikrodenetleyiciler ise TTL uyumludurlar. TTL için lojik-0, 0V ile 2,5V arasında, lojik-1 ise 3V ile 5V arasında tanımlanmaktadır. Dolayısı ile RS232, mikrodenetleyiciler ile doğrudan iletişim sağlayamaz. İletişim kurabilmek için gerilim dönüştürücü devreler veya entegreler kullanmak gerekir. MAX232 ailesi üyelerinin tamamı, harici devrelerin RS-232 portu ile arayüz oluşturması amacıyla üretilmişlerdir. Bu entegre ile bilgisayarın RS232 portundan gelen bilgiler mikrodenetleyiciye uygun TTL seviyesine çevrilirken, mikrodenetleyiciden gelen TTL seviyesindeki bilgileri de RS232'nin tanıyabileceği gerilim seviyelerine

çevirebilmektedir. Şekil 5.29'de MAX232 entegresinin pin yapısı ve iç yapısı görülmektedir.



CAPACITANCE (µF)					
DEVICE	C1	C2	C3	C4	C5
MAX220	0.047	0.33	0.33	0.33	0.33
MAX232	1.0	1.0	1.0	1.0	1.0
MAX232A	0.1	0.1	0.1	0.1	0.1



Şekil 5.20. MAX232 entegresi ve iç yapısı

Entegrenin 2 giriş ve 2 çıkış ucu vardır. RS232 portuna bilgi gönderen (TX) uçlar 7 ve 14 nolu T1OUT, T2OUT uçlarıdır. Bu uçlarda biri RS232 portunun RX ucuna bağlanmalıdır. RS232 portundan bilgi alan (RX) uçlar ise 8 ve 13 nolu R1IN, R2IN uçlarıdır. Bu uçlardan biri de RS232 portunun TX ucuna bağlanmalıdır. Mikrodenetleyiciden veya TTL uyumlu herhangi bir entegreden gelen bilgileri alan uç T1IN veya T2IN, gönderen uç ise R1OUT veya R2OUT uçlarıdır. RS232 DB9 konnektörünün pin yapısı ve açıklamaları Çizelge 5.11'de görülmektedir.

Çizelge 5.11. RS232 portu pin özellikleri

Pin no	Sembol	Açıklama
1	CD	Veri Taşıyıcı Algılayıcı
2	RD-RXD	Veri Alma
3	TD-TXD	Veri Gönderme
4	DTR	Veri Terminal Hazır
5	GND	Toprak
6	DSR	Veri Hazır
7	RTS	Gönderme İsteği
8	CTS	Göndermek için Sil
9	RI	Halka Gösterici

Devre şemasında görülen J5, DB9 konnektörü olup bilgisayarın RS232 portu ile bağlantıyı sağlamaktadır. MAX232 entegresinde bilgisayarla veri alışverişi için 13 ve 14 nolu pinler (R1IN ve T1OUT), PIC ile veri alışverişi için 11 ve 12 nolu pinler (T1IN ve R1OUT) kullanılmıştır.

5.4.3. Güç kaynağı devresi

Güç kaynağı devresi hem mikrodeneteleyicili devreyi, hem seri iletişim devresini hem de kızılötesi yakınlık sensörlerini beslemek için kullanılmaktadır. Güç devresi için 10W gücünde 2x6V çıkışlı bir transformatör (TR1), köprü diyot, 1000uF/63V filtre kondansatörü (C3), ve 7805 (U2) regüle entegresi kullanılmaktadır. 7805 entegresinin çıkışından alınan +5V'luk gerilim hem seri iletişim devresine hem de mikrodenetleyicili kontrol devresine uygulanmaktadır. Ayrıca J3 konnektörünün 21, 22, 24, 25 nolu terminalleri vasıtası ile kızılötesi yakınlık sensörlerini beslemektedir. 7805 entegresinin çıkış uçlarına paralel bağlanan 220Ω'luk R7 direnci ve D1 ledi devrede enerjinin olup olmadığını göstermektedir.

5.5. Mikrodenetleyici Yazılımı

Kullanılan 16F877A mikrodenetleyicisine yüklenecek yazılım CCS C programında yazılmış ve derlenmiştir. CCS (Custom Computer Services Inc.) firması PIC mikrodenetleyicilerinin C dilinde programlanmasını sağlayan tümleşik yazılımlar geliştiren bu alanda deneme kartları üreten bir firmadır. Bu firmanın PIC ürünleri için CCS C adında bir C derleyici programı mevcuttur. CCS C programı PIC10XX, PIC12XX, PIC14XX, PIC16XX, PIC18XX ürünlerini desteklemekte ve 24 bitlik PIC ile dsPIC için de ayrı versiyonları bulunmaktadır. CCS C derleyici içinde bulunan hazır fonksiyonların yanında çevresel birimler ve iletişim protokolleri için hazır birçok kütüphane dosyası bulunmaktadır. Buna ilaveten CCS C'nin kendi forum sitesi ve diğer sitelerde CCS C için yazılmış çok sayıda kütüphane dosyasına rahatlıkla ulaşılabilmektedir. CCS C derleyicisi diğer C derleyicilerine nazaran daha kolay bir şekilde programlama yapmaya imkân vermektedir. Kullanılan denetleyicinin donanım yapısını tam bilmeden bile ileri düzeyde programlar yazılabilmektedir. Bu özellik CCS C derleyicisini diğer C derleyicilerine göre daha popüler yapmıştır. Bahsedilen avantajları nedeniyle CCS C derleyicisi, bu proje için program geliştirmede derleyici olarak tercih edilmiştir.

Aşağıda PIC 16F877A mikrodenetleyicisi için yazılan program akış diyagramları yardımı ile açıklanmış, gerek görülen yerlerde program satırları açıklamaları ile beraber verilmiştir. Program satırlarının tamamı eklerde bulunmaktadır. İki slash (//)'dan sonra yazılanlar derleyici tarafından dikkate alınmaz. Programcı tarafından açıklayıcı olarak yazılan bilgilerdir.

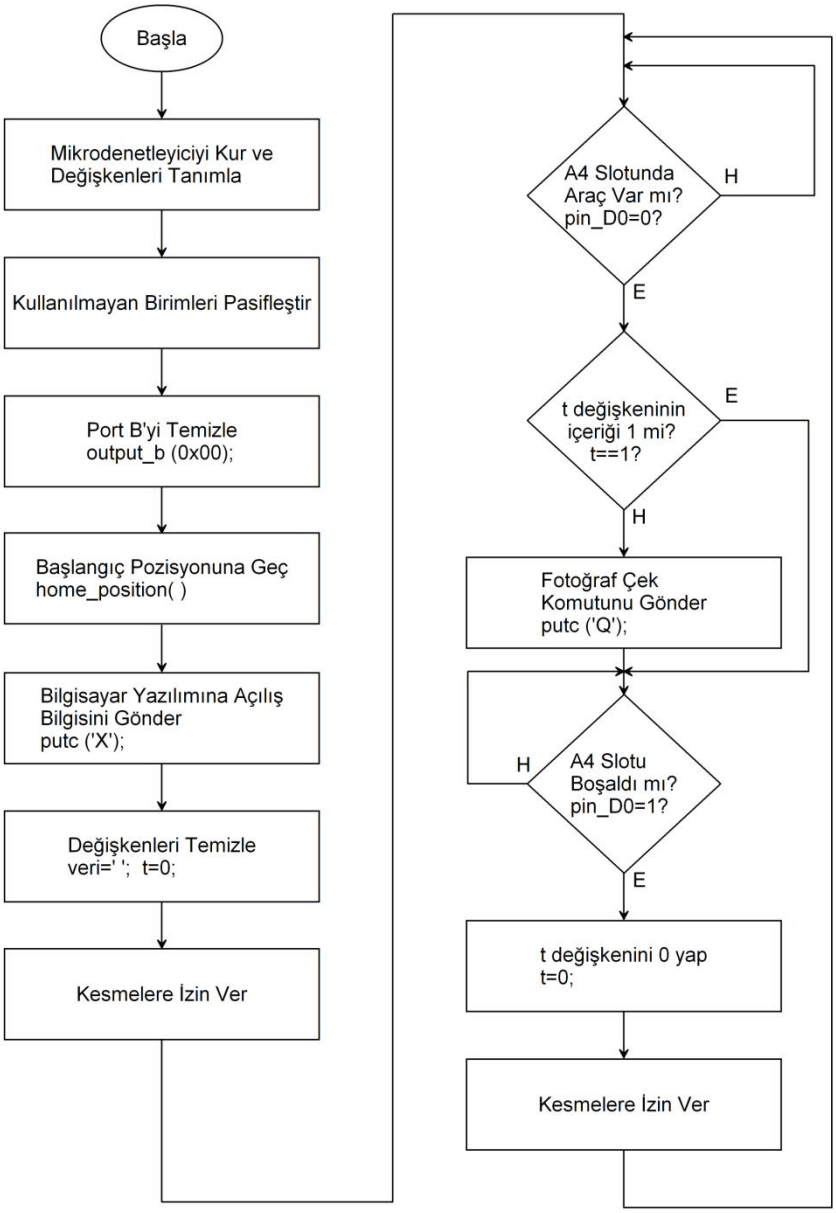
Yazılımın çalışması genel olarak dört kısımdan oluşmaktadır. Bunlar; programın başlatılması, araç girişi işlemleri, araç çıkışı işlemleri ve kesme oluştuğunda gerçekleştirilen işlemlerdir. Bu kısımlar akış diyagramları ve yeri geldikçe program kodları ile açıklanacaktır.

5.5.1. Programın başlatılması

Akış diyagramından da (Şekil 5.22) görüleceği üzere mikrodenetleyicinin kurulması aşamasında önce kullanılacak PIC mikrodenetleyicisi tanıtılıyor. Daha sonra ise konfigürasyon ayarları (fuses) yapılıyor. Kullanılacak osilatör frekansının 4MHz olduğu bildirildikten sonra seri iletişim ayarları tanıtılıyor. Seri iletişimin 9600 baud veri hızında olacağı, C portunun 6 ve 7 nolu pinlerinin veri alışverişi için kullanılacağı bildiriliyor. Son olarak da kullanılacak değişkenler ve kaç bitlik oldukları bildiriliyor. Bu ayarlara ilişkin kod satırları Şekil 5.21'de görülmektedir.

```
#include <16f877a.h>     // Kullanılan mikrodenetleyici
tanıtılıyor.
#fuses XT, NOWDT, NOPROTECT, NOBROWNOUT, NOLVP, NOPUT, NOWRT,
NODEBUG, NOCPD          //Mikrodenetleyicinin konfigürasyon
özellikleri tanıtılıyor.
#use delay(clock=4000000) // 4MHz osilatör frekansı.
#use rs232(baud=9600, xmit=pin_c6, rcv=pin_c7, parity=N, stop=1)
                     // RS232 iletişimi için ayarlar
                     tanıtılıyor.
int16 i;             // 16 bitlik değişken.
int veri;            // 8 bitlik değişken.
int1 t;              // 1 bitlik değişken.
```

Şekil 5.21. Mikrodenetleyicinin kurulmasına ilişkin kod satırları



Şekil 5.22. Mikrodenetleyici programı akış diyagramı

Ana program, mikrodenetleyici kurulduktan sonra programın işlemeye başladığı ilk satırlardır. Öncelikle kullanılan PIC mikrodenetleyicisine ait kullanılmayan özellikler (ADC, analog giriş, zamanlayıcı vb.) pasif yapılır. Hatalı çalışmaya neden olmaması için adım motorlara bilgi sağlayan PORTB temizlenir. “home_position” fonksiyonu ile elektromekanik sistemin başlangıç pozisyonu alması ve iş yapmaya hazır duruma gelmesi sağlanır. Daha sonra bilgisayar yazılımı ile senkronizasyonu sağlamak için “X” bilgisi gönderilir. Bu bilgiyi alan yazılım, ekranda görülen bir metin kutusu ile iletişimin kurulduğunu bildirir. Kullanılacak değişkenler temizlenir ve kesmeler aktif yapılır.

```
void main()                              // ANA PROGRAM FONKSİYONU
   {
   setup_psp(PSP_DISABLED);              // PSP birimi devre dışı.
   setup_spi(SPI_SS_DISABLED);           // SPI birimi devre dışı.
   setup_timer_1(T1_DISABLED);           // T1 zamanlayıcısı pasif.
   setup_timer_2(T2_DISABLED,0,1);       // T2 zamanlayıcısı pasif.
   setup_adc_ports(NO_ANALOGS);          // Analog giriş yok.
   setup_adc(ADC_OFF);                    // ADC birimi devre dışı.
   setup_CCP1(CCP_OFF);                  // CCP1 birimi devre dışı.
   setup_CCP2(CCP_OFF);                  // CCP2 birimi devre dışı.
   output_b(0x00);       // PortB'yi temizle. (adım motorlar)
   home_position();                      // Başlangıç konumuna geç.
   delay_ms(500);                        // 0,5 Sn bekle.
   putc('X');            //Bilg.Programına başla komutunu gönder.
   delay_ms(500);                        // 0,5 Sn bekle.
   veri=' ';             // "veri" değişkenini temizle.
   t=0;                                  // "t" değişkenini 0 yap.
   enable_interrupts(GLOBAL);            // Tüm kesmelere izin ver.
   enable_interrupts(int_rda);     // RS232 kesmesine izin ver.
```

Şekil 5.23. Mikrodenetleyici kurulduktan sonra ana programın başladığı kod satırları

Buraya kadar yapılan işlemler, elektromekanik sistemin araç kabulüne veya teslimine hazır duruma gelmesini ve bilgisayar yazılımı ile senkronizasyonun sağlanmasını amaçlamaktadır.

Burada “home_position ()” fonksiyonun çalışmasının açıklanmasında yarar görülmektedir. Fonksiyon çağırıldığında Şekil 5.24’de görülen satırlar PIC tarafından işleme alınır.

```
void home_position()    //Başlangıç pozisyonuna gelme fonksiyonu.
   {
   while (input(pin_a2)==1)
      {
         output_low(pin_b0);    //Yukarı/Aşağı motor için yön
output_low(pin_b1);
         delay_us(750);
         output_high(pin_b1);
         delay_us(750);
      }
   output_high(pin_b2);    // Sağ/Sol motor için yön bilgisi.
   do                      // RA3 0 olana kadar döngü devam eder.
      {
         output_high(pin_b3);
         delay_us(750);
         output_low(pin_b3);
         delay_us(750);
      }
   while (input(pin_a3)==1);
   output_low(pin_b2); //Sağ/Sol hareketi sağlayan motor için yön
   for (i=0;i<=7822;i++)    // i<= olduğu sürece döngü devam eder
      {
         output_high(pin_b3);
         delay_us(750);
         output_low(pin_b3);
         delay_us(750);
      }
   output_high(pin_b0);    // Yukarı/Aşağı motor için yön
   for (i=0;i<=302;i++)    // i<=302 olduğu sürece döngü
      {
         output_high(pin_b1);
         delay_us(750);
         output_low(pin_b1);
         delay_us(750);
      }
   }
```

Şekil 5.24. Başlangıç pozisyonuna gelme fonksiyonuna ilişkin kod satırları

Şekil 5.24'de görülen satırlar home_position fonksiyonuna aittir. Bu fonksiyonun işlevi sistemi, ilk açıldığında araç kabulüne veya teslim edilmesine hazır pozisyona getirmektir. Buna başlangıç pozisyonu da denilebilir. Başlangıç pozisyonuna geçmek için paletin konumu ayarlanmalıdır. Bu da mekanik anahtarlar yardımı ile olmaktadır. Öncelikle alt mekanik anahtarın (alt switch – sw2) kapalı olması gerekmektedir. Kapalı değil ise dikey hareketi sağlayan motor paleti aşağı indirerek bu anahtarın kapanması sağlanır. Alt anahtar kapandıktan sonra yatay hareketi sağlayan motor yardımı ile palet sağa dönmektedir. Bu dönüş yan mekanik anahtar (yan switch – sw1) kapanana kadar devam etmektedir. Yan anahtar da kapandıktan sonra palet platformun tam ortasına gelmelidir. Bunun için sola doğru 110^{o}'lik bir dönüş yapmalıdır. Bu dönüş için gerekli adım sayısı 7822'dir. 7822 adımlık bir sola dönüş gerçekleştirilerek palet 110^{o} dönüş yapar ve platformun tam ortasına gelir. Yatay hareketi sağlayan motorun bir tam dönüşü sürücü üzerindeki anahtarların konumu ile 25600'e ayarlanmıştır. Yani 360°'lik bir tam tur 25600 adımda gerçekleşmektedir. 110^{o}'lik bir dönüş için gerekli hesaplamalar Eş. 5.38 ve Eş. 5.39'da görülmektedir.

$$1° \; için \; adım \; sayısı = \frac{25600}{360} = 71.11 \tag{5.38}$$

$$110° \; için \; gerekli \; adım \; sayısı = 71.11 * 110 = 7822.1 \tag{5.39}$$

Başlangıç pozisyonunun tam olarak tamamlanması için paletin yaklaşık 17mm yukarı çıkması gerekmektedir. Bunun için yukarı aşağı hareket eden motor, paleti yukarı çıkarmak için 302 adım dönmelidir. Dikey hareketi sağlayan motorun bir tam dönüşü sürücü üzerindeki anahtarların konumu ile 1600 adıma ayarlanmıştır. Yani 360^{0}'lik bir tam tur 1600 adımda gerçekleşmektedir. Motor ucuna takılı olan dişlinin diş üstü çapı 28,65mm olduğuna göre 17mm yukarı hareket için gerekli hesaplamalar Eş. 5.40 - 5.44'de görülmektedir.

$$Dişli \; çevresi = 2\pi r = 2 * 3.14 * 14.325 = 89.961mm \tag{5.40}$$

$$1° \; dönüş = \frac{89,961}{360} \cong 0.2499mm \tag{5.41}$$

$$17mm\ için\ gerekli\ dönüş\ açısı = \frac{17}{0.2499} \cong 68{,}02°\ \ 68°\ alınmıştır \quad (5.42)$$

$$1°\ için\ adım\ sayısı = \frac{1600}{360} \cong 4.444 \quad (5.43)$$

$$68°\ için\ gerekli\ adım\ sayısı = 68 * 4.444 \cong 302 \quad (5.44)$$

Yukarı/aşağı hareketi sağlayan motorun dönüş yönü RB0 portundaki bilgi ile kontrol edilmekte ve bu porta 1 bilgisi uygulandığında saat yönünde dönerek yukarı hareketi sağlamaktadır. Eğer 0 bilgisi uygulanırsa saat yönünün tersinde dönerek aşağı hareketi sağlamaktadır. Aynı motorun dönüşü için gerekli clock palsi RB1 portuna uygulanmaktadır. Clock palslerinin 0 ve 1 (low ve high) geçişleri arasına 750µsn'lik gecikme süreleri konulmuştur.

Sağ/sol dönüşü sağlayan motorun dönüş yönü RB2 portundaki bilgi ile kontrol edilmekte ve bu porta 1 bilgisi uygulandığında saat yönünde dönerek sağa dönüş hareketi sağlamaktadır. Eğer 0 bilgisi uygulanırsa saat yönünün tersinde dönerek sola dönüş hareketi sağlamaktadır. Aynı motorun dönüşü için gerekli clock palsi RB3 portuna uygulanmaktadır. Clock palslerinin 0 ve 1 (low ve high) geçişleri arasına 750µsn'lik gecikme süreleri konulmuştur.

5.5.2. Araç kabulu veya teslimi işlemleri

Araç kabulünde veya tesliminde asıl işlemler, sonraki bölümde anlatılacak olan kesme fonksiyonu içerisinde yapılmaktadır. Şekil 5.25'deki kod satırları aracın girişini algılayan, plaka tanımlaması için fotoğraf çekilmesi komutunu yollayan ve aracın çıkışını algılayan işlemleri anlatmaktadır.

```
kontrol:
   do
   {
      delay_ms(100);
   }
   while (input(PIN_D0)==1); //A4'e araç gelene kadar bekle
   delay_us(500);            // 0,5 mSn bekle.

   if (t==1)                 // "t" değişkeni 1 ise;
      {
         goto bekle; //Araç teslimi yapıldı,"bekle" etiketine git
      }
   putc('Q');       // Araç girişi var, foto çek komutu gönder.
   delay_ms(200);           // 0,2 Sn bekle.
bekle:
 do
      {
      }
   while (input(PIN_D0)==0);   // A4 dolu olduğu surece bekle.
   t=0;                        // "t" değişkenini 0 yap.
   enable_interrupts(GLOBAL);  // Tüm kesmelere izin ver.
   enable_interrupts(int_rda); // RS232 kesmesine izin ver.
   delay_ms(3000);             // 3 Sn bekle.
   goto kontrol;               // "kontrol" etiketine git.
   }
```

Şekil 5.25. Araç giriş ve çıkışını algılayan kod satırları

"kontrol" etiketinin hemen altındaki *do…while* döngüsü ile A4 bölmesine araç girişi olup olmadığı kontrol edilir. Herhangi bir şekilde araç girişi yoksa veya RS232 kesmesi (int_rda) olmadığı sürece döngü devam eder.

Araç girişi olduğunda döngüden çıkılır ve "t" değikeni kontrol edilir. Eğer "t" değişkeni 1 değil ise bilgisayar programına "Q" bilgisi gönderilir. Program bu bilgiyi aldığında aracın fotoğrafını çeker ve plakasını tanımlar.

Plaka tanımlama işlemi tamamlanıp bilgisayar programından bilgi gelene kadar “bekle” etiketi altındaki *do…while* döngüsüne girilir. Döngü, araç A4 bölmesinden alınana kadar devam eder. PIC programı bu döngü içerisinde iken, plakayı tanımlayan bilgisayar programından park etmek için “W” bilgisi gelince kesme oluşur. Program kesmeye giderek park etme işlemini tamamlar. Kesmeden çıkıp kaldığı yerden (döngü içi) devam eden program, A4 bölmesi boşaldığı için döngüden çıkar. “t” değişkeni temizlenir ve kesmelere yeniden izin verilir. Çünkü kesme işlemine gidildiğinde *int_rda* kesmesi pasif edilmiştir. Program tekrar “kontrol” etiketine yönlendirilerek sonraki işlem için hazır duruma geçmesi sağlanır. Programın ikinci kez *do…while* döngüsüne girmesindeki amaç park etmek üzere “W” bilgisi gelene kadar bekletmektir. Aksi durumda, araç A4 bölmesinde iken program sürekli işlemekte, A4 bölmesinde araç olduğu için birinci *do…while* döngüsünden çıkmakta ve defalarca “Q” bilgisi göndererek olumsuzluklara neden olmaktadır.

Program birinci *do…while* döngüsünde iken, bilgisayar programı tarafından teslim edilecek araç var bilgisi gelirse (A, B, C, D, E, F, G, H, I, J, K, L, M) yine kesme oluşur. Program kesme fonksiyonuna giderek, gelen bilgiye göre, aracın bölmesinden alınıp A4 bölmesine konulmasını sağlar. Burada kesmeden çıkılmadan önce “t” değişkenine 1 bilgisi yüklendiğine dikkat edilmelidir. Kesme işlemi bitip döngü içine tekrar gelindiğinde A4 bölmesinde araç olduğu için birinci döngüden çıkılır ve “t” değişkeni kontrol edilir. Bu sefer 1 olduğu için “bekle” etiketine gidilerek araç A4 bölmesinden alınana kadar *do…while* döngüsü içinde bekler. Araç alındığında, A4 bölmesi boşaldığı için döngüden çıkar. “t” değişkeni temizlenir ve kesmelere yeniden izin verilir. Program tekrar “kontrol” etiketine yönlendirilerek sonraki işlem için hazır duruma geçmesi sağlanır.

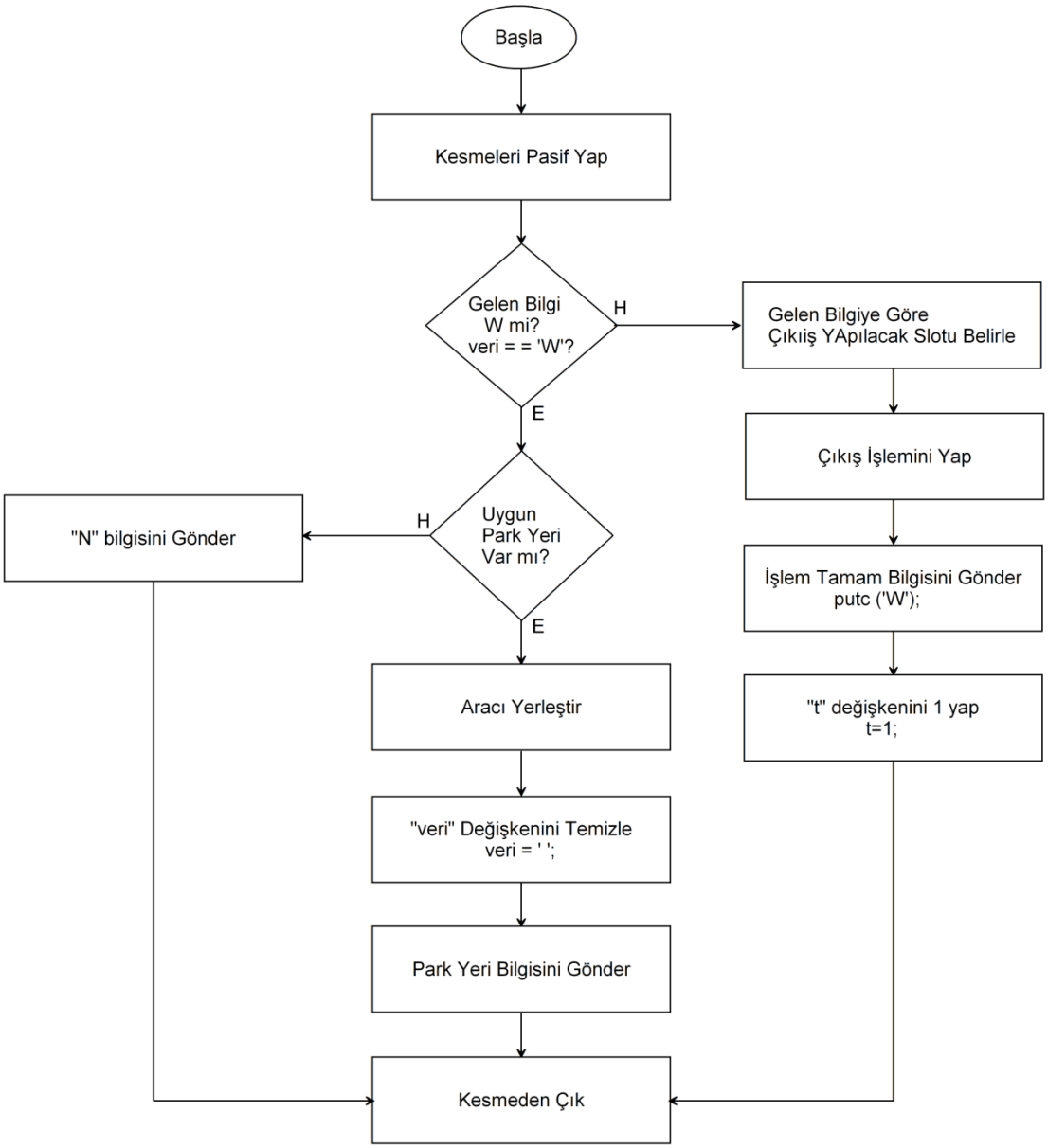
Araç teslimi işleminde “t” değişeninin 1 yapılmasındaki amaç; programı “bekle” etiketine yönlendirerek “Q” komutunun gönderilmesini önlemektir. Çünkü bu sefer gelen araç yoktur, teslim edilen araç vardır. Dolayısı ile fotoğraf çekme işlemi de gereksizdir. Ayrıca gereken yerlerde *delay* komutu ile gecikmeler kullanılmıştır.

5.5.3. Kesme fonksiyonu

Kesme (interrupt) fonksiyonu, programın çalışması esnasında kesme oluştuğunda işlenmesi istenilen kodların bulunduğu fonksiyondur. Kesme işlemi kullanıcıya program geliştirmede büyük kolaylıklar sağlamaktadır. Program normal akışında devam ederken kesme oluştuğunda, program son işlediği satırı kaydederek, kesme fonksiyonun içeriğindeki satırları işler. Kesme fonksiyonundan çıktığında ise daha önce kaldığı yerden programın işleyişine devam eder. Program içinde kesmenin sürekli kontrol edilmesine gerek yoktur. Örneğin seri iletişim kesmesi göz önüne alınırsa, eğer kesme olmasaydı program çalışırken seri iletişim kanalını sürekli olarak kontrol etmek gerekecekti. Bu durum hem programın işleyişini yavaşlatacak hem de gereksiz program satırlarının yazılmasına neden olacaktı.

“#int_rda” kesmesi mikrodenetleyicinin seri haberleşme veri alma pinine (RX) bilgi geldiği zaman aktif olur. Seri iletişim kesmesinde, kesmeye girildikten sonra kesme pasif edilmelidir. Aksi halde program hep kesmeye gidecektir.

Bu uygulamada kesme fonksiyonu, aracın kabulü veya teslimi esnasında mekanik sistemin hareketlerinin kontrol edildiği satırları barındırmaktadır. Şekil 5.26’da kesme fonksiyonunda gerçekleştirilen işlemlerin akış diyagramı görülmektedir.



Şekil 5.26. Kesme fonksiyonu akış diyagramı

Yapılan çalışmada bilgisayar programı kullanıldığı ve bu program ile PIC entegresi ile RS232 iletişim protokolünü kullanarak haberleştiği belirtilmişti. Bilgisayar programı ise bir sonraki bölümde detayları ile açıklanacaktır. Bilgisayar programından PIC'e seri iletişim kanalı ile bilgi geldiğinde seri iletişim kesmesi (int_rda) oluşur. Kesme oluştuğunda da "rs232_kesmesi" fonksiyonu çalışır.

Kesme fonksiyonunu çalıştığında ilk olarak *int_rda* kesmesi pasif edilerek kesmenin tekrarlanması önlenir. Daha sonra bilgisayardan gelen bilgi okunarak "veri" kaydedicisine kaydedilir. Eğer gelen veri "W" ise araç kabul bölmesi olan A4 bölmesinde park etmek için bir aracın beklediği anlaşılır.

En uygun (en kısa yoldan ve en az enerji harcanarak) boş park yeri belirlenir ve aracın park edilmesi için ilgili fonksiyon çalıştırılır (örneğin a4den_a5e_koy). Daha sonra aracın park edildiği bölmeye ait bilgi bilgisayar programına gönderilir (örneğin D). Bu bilgi, park edilen araç teslim edilirken geri çağırma işleminde kullanılacaktır. Bu bilgi gelene kadar bilgisayar programı başka bir işlem yapmaz, bekler. Aracın park edildiği bölmeye ilişkin bilgi bilgisayar programı tarafından alındığında araç için park süresi de başlatılır. Eğer boş park yeri yoksa bilgisayar programına "N" bilgisi gönderilir.

Eğer gelen bilgi "W" değil ise daha önce park edilmiş araçlardan birinin yerini bildiren bir bilgidir (A, B, C, D, E, F, G, H, I, J, K, L, M) ve araç teslim edilecektir. Bu durumda gelen bilgiye göre bölmesinde bekleyen araç ilgili fonksiyon çalıştırılarak (örneğin A5den_A4e_koy) alınıp A4 bölmesine yerleştirilir. Daha sonra teslim etme işleminin bittiğini bilgisayar programına bildirmek için PIC tarafından "W" bilgisi gönderilir. Bilgisayar programı bu veriyi alana kadar başka bir işlem yapmaz, bekler. Sonraki bölümde de anlatılacağı üzere bilgisayar programı park süresine göre ücret hesaplaması yapmakta aynı zamanda o aracın daha önce kaç kez park için geldiğinin kaydını da tutmaktadır. Son olarak da "t" değişkenine 1 bilgisi yüklenir.

Park etmek için araç geldiğinde PIC'e "W" bilgisi gönderilmeden önce aracın fotoğrafı çekilir ve bilgisayar programı tarafından plaka tanımlanması yapılır. Bu işlemler de yine sonraki bölümde anlatılacaktır. Çizelge 5.12'de hangi park yeri için hangi bilginin kullanıldığı görülmektedir.

Çizelge 5.12. Park bölmesine göre kullanılan bilgi

Park Bölmesi	A1	A2	A3	A5	A6	A7	B1	B2	B3	B4	B5	B6	B7
Gönderilen Bilgi	A	B	C	D	E	F	G	H	I	J	K	L	M

Kesme fonksiyonunun içeriğinde işleyen program kodları ek-1'de verilen mikrodenetleyici yazılımında incelenebilir.

Araç park edilirken veya teslim edilirken, otopark iki katlı olduğu için bir kat yukarı çıkılması veya yukarıda ise bir kat aşağı inilmesi, aracın üzerinde bulunduğu paletin ileri ya da geri hareket ettirilmesi, palet aracın altına girdiğinde aracın tekerleklerini yerden kesilmesi için kaldırılması, palet üzerindeki aracın bölmeye konulması esnasında paletin indirilmesi gerekmektedir. Bahsedilen işlemler için yazılmış fonksiyonlar vardır. Programın çalışmasının tam olarak anlaşılması için bu fonksiyonların incelenmesi gerekmektedir.

```
void kat_cik()           // Bir kat yukarı çıkma fonksiyonu.
   {
   output_high(pin_b0);  // Yukarı/Aşağı hareket motoru için yön
   for (i=0;i<=3555;i++) // i<=3555 olduğu sürece döngü
   // Bu döngü bittiğinde palet bir üst kata çıkar.
      {
         output_high(pin_b1);
         delay_us(750);
         output_low(pin_b1);
         delay_us(750);
      }
   }
```

Şekil 5.27. Bir kat yukarı çıkma fonksiyonuna ilişkin kod satırları

Tasarlanan otoparkın iki katlı olduğu önceki bölümlerde belirtilmişti. Bir kat yukarı çıkmak için gerekli "kat_çık" fonksiyonuna ait satırlar Şekil 5.27'de görülmektedir. Katlar arası mesafe 200mm'dir. Daha önce 1^o'lik dönüşün yaklaşık 0.2499 mm hareket sağladığı ve 1^o dönüş için gerekli adım sayısının 4,444 olduğu hesaplanmıştı (Bkz. Eş. 5.41 ve Eş. 5.43). 200mm için gerekli adım sayısı ise 3555'dir. İlgili hesaplamalar Eş. 5.45 ve Eş. 5.46'da görülmektedir.

$$200mm\ için\ gerekli\ dönüş\ açısı = \frac{200}{0.2499} \cong 800° \tag{5.45}$$

$$800°için\ gerekli\ adım\ sayısı = 800 * 4.444 \cong 3555\ adım \tag{5.46}$$

```
void kat_in()              // Bir kat aşağı inme fonksiyonu.
   {
   output_low(pin_b0);     // Yukarı/Aşağı hareketi sağlayan
motor için yön bilgisi.
   for (i=0;i<=3555;i++)   // i<=3555 olduğu sürece döngü
   // Bu döngü bittiğnde palet bir alt kata iner.
      {
         output_high(pin_b1);
         delay_us(750);
         output_low(pin_b1);
         delay_us(750);
      }
   }
```

Şekil 5.28.Bir kat aşağı inme fonksiyonuna ilişkin kod satırları

Bir kat aşağı inmek için “kat_in” fonksiyonu kullanılmaktadır. Kat çıkmak için yapılan hesaplamalar kat inmek için de geçerlidir. Tek fark motor dönüş yönünün değişmesidir. İlgili kod satırları Şekil 5.28’de incelenebilir.

```
void ileri()  //Paletin ileri hareket etmesini sağlayan fonksiyon
   {
   output_high(pin_b4);    // İleri/Geri hareket motoru için yön
   do                      // RA0 0 olana kadar döngü devam eder.
                        // Uçtaki röle kapanana kadar ileri gider
      {
         output_high(pin_b5);
         delay_us(750);
         output_low(pin_b5);
         delay_us(750);
      }
   while (input(pin_a0)==1);
   }
```

Şekil 5.29. Paletin ileri hareketine ilişkin kod satırları

Paletin ileri hareket ederek (uzayarak) aracın altına girmesini sağlayan fonksiyon “ileri” fonksiyonudur. Ne kadar ileri hareket edeceği hesaplamalar ile belirlenmemiş olup manyetik bir röle ile kontrol edilmektedir. Paletin yan tarafında bulunan mıknatıs, uçtaki manyetik rölenin kontaklarını kapatana kadar ileri doğru hareket devam eder. Manyetik rölenin kontakları kapandığında paletin hareketi durur. Şekil 5.29’da görülen kodlar bu işlemin yapılmasına ait program satırlarıdır. Öncelikle ileri/geri hareketi sağlayan motorun yön kontrolü ayarlanarak ileri hareket için hazır duruma getirilir. Daha sonra RA0 portuna bağlı kontak kapanana kadar motorun dönüşü için gerekli clock sinyalleri uygulanır. Kontak kapandığında mikrodenetleyiciye 0 bilgisi ulaşır ve ileri hareket sona erer.

İleri/geri hareketi sağlayan motorun dönüş yönü RB4 portundaki bilgi ile kontrol edilmekte ve bu porta 1 bilgisi uygulandığında saat yönünün tersinde dönerek ileri hareketi sağlamaktadır. Eğer 0 bilgisi uygulanırsa saat yönünde dönerek sola geri hareketi sağlamaktadır. Aynı motorun dönüşü için gerekli clock palsi RB5 portuna uygulanmaktadır. Clock palslerinin 0 ve 1 (low ve high) geçişleri arasına 750µsn’lik gecikme süreleri konulmuştur.

```
void geri()  // Paletin geri hareket etmesini sağlayan fonksiyon.
   {
   output_low(pin_b4);     // İleri/Geri hareketi sağlayan motor
için yön bilgisi.
   do                      // RA1 0 olana kadar döngü devam eder.
                  // Gerideki reed röle kapanana kadar geri gelir
      {
         output_high(pin_b5);
         delay_us(750);
         output_low(pin_b5);
         delay_us(750);
      }
   while (input(pin_a1)==1);
   }
```

Şekil 5.30. Paletin geri hareketine ilişkin kod satırları

Palet aracın altına girip araç kaldırıldıktan sonra bölme içerisinden çıkarılması için paletin geri hareket etmesi gerekmektedir. Bu işlem “geri” fonksiyonu ile gerçekleştirilmektedir. İleri hareketi sağlayan fonksiyon ile benzer olup motor dönüş yönü değişmekte ve benzer mantıkla RA1 portuna bağlı diğer bir manyetik rölenin kapanması gerekmektedir. İlgili kod satırları Şekil 5.30’da incelenebilir.

```
void kaldir()   // Palet aracın altına girdikten sonra
                // aracın yukarı kaldırmasını sağlayan fonksiyon
   {
   output_high(pin_b0);    // Yukarı/Aşağı hareket motoru için
yön
   for (i=0;i<=356;i++)    // i<=356 olduğu sürece döngü
                  // Döngü bittiğinde araç yukarı kaldırılmıştır.
      {
         output_high(pin_b1);
         delay_us(750);
         output_low(pin_b1);
         delay_us(750);
      }
   }
```

Şekil 5.31. Aracın bir miktar yukarı kaldırılmasına ilişkin kod satırları

Palet aracın altına girdikten sonra aracı bölme dışına çıkarabilmesi için tekerleklerinin yer ile temasının kesilmesi gerekir. Yani aracın bir miktar yukarı kaldırılması gerekmektedir. Kaldırma işlemine ilişkin kod satırları Şekil 5.31’de görülmektedir. Bu işlem için “kaldır” fonksiyonu kullanılarak aracın 20mm yukarı kaldırılması sağlanmaktadır. Yukarı kaldırma işlemi yukarı/aşağı hareketi sağlayan motor hareketi ile gerçekleşmektedir. Daha önce 1^o’lik dönüşün yaklaşık 0.2499 mm hareket sağladığı ve 1^o dönüş için gerekli adım sayısının 4,444 olduğu hesaplanmıştı (Bkz. Eş. 5.41 ve Eş. 5.43). 20mm için gerekli adım sayısı ise yaklaşık 356’dir. İlgili hesaplamalar Eş. 5.47 ve Eş. 5.48’de görülmektedir.

$$20mm\ için\ gerekli\ dönüş\ açısı = \frac{20}{0.2499} \cong 80° \tag{5.47}$$

$$80°için\ gerekli\ adım\ sayısı = 800 * 4.444 \cong 356\ adım \quad (5.48)$$

```
void indir()     // Aracın yere indirilmesini sağlayan fonksiyon
   {
   output_low(pin_b0);  // Yukarı/Aşağı hareket motoru için yön
   for (i=0;i<=356;i++)    // i<=356 olduğu sürece döngü
                    // Döngü bittiğinde araç aşağı indirilmiştir
      {
         output_high(pin_b1);
         delay_us(750);
         output_low(pin_b1);
         delay_us(750);
      }
   }
```

Şekil 5.32. Aracın yere indirilmesine ilişkin kod satırları

Araç yerleştirileceği bölme içerisine geldiğinde yere indirilmesi gerekmektedir. Bu işlem için "indir" fonksiyonu kullanılmaktadır. Aracı kaldırmak için yapılan hesaplamalar yere indirmek için de geçerlidir. Tek fark motor dönüş yönünün değişmesidir. Kod satırları Şekil 5.32'de incelenebilir.

Aracın kabul bölmesinden alınıp diğer bölmelere park edilmesinde veya park edilmiş aracın ilgili bölmeden alınıp kabul bölmesinden teslim edilmesinde paletin yatayda bir bölmeden diğer bölmeye (sağ/sol hareket) hareket etmesi gerekmektedir. Otopark bölmelerinin yarım daire (180^o) şeklinde tasarlandığı ve bir katta 7 bölme olduğu önceki bölümlerde anlatılmıştır. Dolayısı ile bir bölmeden komşu bölmeye hareket için $180^o/7=25.7^o$'lik bir hareket gerekmektedir. Bu hareket için gerekli adım sayısı ise $25.7^o*71.11=1827$'dir. Fakat tasarımın el işçiliği ile yapılması nedeni ile 1^o-2^o'lik farklılıklar olabilmektedir. O nedenle bir bölmelik hareket için bazen 1800 adım bazen 1900 adım, 2 bölmelik hareket için bazen 3700 adım bazen 3800 adım, 3 bölmelik hareket için ise 5600 adım hareket kullanılmıştır.

“pulse_1800”, “pulse_1900”, “pulse_3700”, “pulse_3800” ve “pulse_5600” fonksiyonları bölmeler arası yatay harekette gerekli adım sayılarını sağlamaktadırlar. Daha önce bahsedildiği gibi yatay hareketi sağlayan motorun yön bilgisi RB2 portuna, dönüş hareketi için gerekli clock palsleri de RB3 portuna uygulanmaktadır. Bahsedilen fonksiyonlar dönüş yönü bilgisi içermemekte, dönüş yönü belirlendikten sonra kullanılmaktadırlar.

```
void pulse_1900 ()          // 1900 Clock palsi sağlayan fonksiyon
   {
   for (i=0;i<=1900;i++)    // i<=1900 olduğu sürece döngü
       {
         output_high(pin_b3);
         delay_us(750);
         output_low(pin_b3);
         delay_us(750);
       }
   }
```

Şekil 5.33. 1900 clock palsi sağlayan kod satırları

5.5.4. Park edilecek en uygun bölmenin tespit edilmesi

En uygun bölmenin seçilmesinde dikkate alınacak kriterler harcanan enerji ve zamandır. Aracın park edilmesinde üç step motor görev yapmaktadır. Motorların hepsi DC 24V gerilim ile sürüldüğü için uygunluk fonksiyonunun oluşturulmasında gerilim bir değişken olarak dikkate alınmamıştır.

Park edilecek en uygun bölme tespit edilirken takip edilecek alternaif güzergâhların olup olmadığı incelenmiştir. Örneğin araç A4 bölmesinden alınıp B1 bölmesine yerleştirlecek ise takip edilebilecek iki alternatif yol vardır. Birinci alternatife göre elektromekanik sistem önce bir kat yukarı çıkıp sonra üç bölmelik mesafe katetmek üzere sola dönmelidir. İkinci alternatife göre ise önce üç bölme sola dönmeli ve sonra bir kat yukarı çıkılmalıdır. Aslında iki farklı alternatif güzergâh varmış gibi görünse de her ikisindede yapılan işlem, harcanan süre ve

enerji aynıdır. Sonuç olarak herhangi bir park bölmesine hareket esnasında avantaj sağlayan alternatif bir güzergâhın olmadığı tespit edilmiştir.

Dikey hareket için kullanılan motor M_1, çektiği akım I_1; yatay hareketi sağlayan motor M_2 çektiği akım I_2'dir. İleri geri hareketi sağlayan motor her durumda aynı hareketi yaptığı için kriter olarak dikkate alınmamaktadır.

M_1 motorunun enerji altında hareket etmez iken çektiği akım 0,25A, yukarı çıkarken çektiği akım 0,54A ve aşağı inerken çektiği akım 0.39A'dir. M_1 motorunun yukarı çıkarken hareketsiz durumuna göre akım farkı Eş. 5.49'da görüldüğü üzere 0,29A, aşağı inerken hareketsiz durumuna göre akım farkı ise Eş. 5.50'de görüldüğü üzere 0,14A'dir. Yukarı çıkış ve aşağı iniş süreleri eşit olduğu için her bir yukarı/aşağı hareket için bu iki akım değerinin ortalaması alınırsa 0,215A bulunur (Eş. 5.51). Bu değer hesaplamalarda I_1 değeri olarak kullanılacaktır.

$$I_{yukarı\ yukarı\ çıkarken\ akım\ farkı} = 0{,}54 - 0{,}25 = 0{,}29A \tag{5.49}$$

$$I_{aşağı\ aşağı\ inerken\ akım\ farkı} = 0{,}39 - 0{,}25 = 0{,}14A \tag{5.50}$$

$$I_1 = (0{,}29 + 0{,}14)/2 = 0{,}215A \tag{5.51}$$

Benzer durum M_2 motoru için de geçerlidir. M_2 motorunun enerji altında hareket etmez iken çektiği akım 0,42A, sağa yada sola dönerken çektiği akım 0,75A'dir. M_2 motorunun sağa yada sola dönerken hareketsiz durumuna göre akım farkı Eş. 5.52'de görüldüğü üzere 0,33A'dir. Araç park edilirken sağa ve sola hareket süreleri eşit olduğu için her yatay hareket için bu akım değeri dikkate alınacaktır. Bu değer hesaplamalarda I_2 değeri olarak kullanılacaktır.

$$I_2 = 0{,}75 - 0{,}42 = 0{,}33A \tag{5.52}$$

Hareket esnasında bir kat çıkış 5,2sn, bir kat iniş 5,2sn, bir bölmelik sağa hareket 2,75sn ve bir bölmelik sola hareket 2,75sn sürmektedir. Kat çıkıp inme süresi toplam 10,4sn olup bu değer t_1 ile ifade edilirken, bir bölmelik sağa/sola hareket toplam 5,5sn olup t_2 ile ifade edilmektedir.

Otoparkın birinci katındaki park alanları A, ikinci katındaki park alanları B olarak adlandırılmıştır. Park Bölmeleri Çizelge 5.13'de ki gibidir.

Çizelge 5.13. Katlara göre park bölmesi isimleri

Kat Numarası	Bölme İsimleri						
2. Kat	B1	B2	B3	B4	B5	B6	B7
1. Kat	A1	A2	A3	A4	A5	A6	A7

A4 bölmesi araç kabul bölmesi olup araç buradan alınıp yine buradan teslim edilmektedir. Bu bölme park alanı olarak kullanılmamaktadır. Tüm hesaplamalar bu bölme dikkate alınarak yapılmaktadır. Bu bilgilere göre A4 bölmesinden diğer bölmelere aracın yerleştirilmesi için gerekli süreler Çizelge 5.14'de görülmektedir.

Çizelge 5.14. Aracın kabul bölmesinden (A4) diğer bölmeler yerleştirilirken geçen süreler

Aracın Alındığı Bölme	Aracın Yerleştirildiği Bölme	Harcanan Süre Formülü	Harcanan Süre (sn)
A4	A1	$0.t_1+3.t_2$	16,5
A4	A2	$0.t_1+2.t_2$	11
A4	A3	$0.t_1+1.t_2$	5,5
A4	A5	$0.t_1+1.t_2$	5,5
A4	A6	$0.t_1+2.t_2$	11
A4	A7	$0.t_1+3.t_2$	16,5
A4	B1	$1.t_1+3.t_2$	26,9
A4	B2	$1.t_1+2.t_2$	21,4
A4	B3	$1.t_1+1.t_2$	15,9
A4	B4	$1.t_1+0.t_2$	10,4
A4	B5	$1.t_1+1.t_2$	15,9
A4	B6	$1.t_1+2.t_2$	21,4
A4	B7	$1.t_1+3.t_2$	26,9

Süre dikkate alındığında A4 bölmesine göre en kısa zamanda park edilme sırası A3, A5, B4, A2, A6, B3, B5, A1, A7, B2, B6, B1, B7 olmaktadır. Sürelerin eşit olduğu durumda numarası küçük olan bölmeye öncelik verilmiştir. Bunun bir avantajı yoktur, tasarımcının tercihine kalmıştır.

Park etme esnasında harcanan enerjiye göre hesaplama yapılırsa Çizelge 5.15'deki bilgiler ortaya çıkacaktır.

Çizelge 5.15. Aracın kabul bölmesinden (A4) diğer bölmeler yerleştirilirken harcanan enerji

Aracın Alındığı Bölme	Aracın Yerleştirildiği Bölme	Harcanan Enerji Formülü	Harcanan Enerji (Watt.sn)
A4	A1	$24.(0.t_1.I_1+3.t_2.I_2)$	130,68
A4	A2	$24.(0.t_1.I_1+2.t_2.I_2)$	87,12
A4	A3	$24.(0.t_1.I_1+1.t_2.I_2)$	43,56
A4	A5	$24.(0.t_1.I_1+1.t_2.I_2)$	43,56
A4	A6	$24.(0.t_1.I_1+2.t_2.I_2)$	87,12
A4	A7	$24.(0.t_1.I_1+3.t_2.I_2)$	130,68
A4	B1	$24.(1.t_1.I_1+3.t_2.I_2)$	184,344
A4	B2	$24.(1.t_1.I_1+2.t_2.I_2)$	140,784
A4	B3	$24.(1.t_1.I_1+1.t_2.I_2)$	97,224
A4	B4	$24.(1.t_1.I_1+0.t_2.I_2)$	53,664
A4	B5	$24.(1.t_1.I_1+1.t_2.I_2)$	97,224
A4	B6	$24.(1.t_1.I_1+2.t_2.I_2)$	140,784
A4	B7	$24.(1.t_1.I_1+3.t_2.I_2)$	184,344

Harcanan enerji dikkate alındığında A4 bölmesine göre en az enerji sarf edilerek park edilme sırası da yine A3, A5, B4, A2, A6, B3, B5, A1, A7, B2, B6, B1, B7 şeklinde olmaktadır. Eşitliğin olduğu durumlarda numarası küçük olan bölmeye öncelik verilmiştir.

Park edilme sırası uygunluk fonksiyonu ile ifade edilmek istenirse fonksiyon Eş. 5.53'deki gibi olur gibi olur.

$$f(k1,k2) = \frac{1}{k_1.t_1.I_1+k_2.t_2.I_2} \tag{5.53}$$

Bu formülde k_1 dikeyde çıkılan kat sayısını, k_2 ise yatayda kaç bölmelik yol alındığını gösterir. Fonksiyonun değeri büyüdükçe harcanan enerji azalacaktır. Eş. durumunda yine numarası küçük olan bölmeye öncelik verilecektir. Araç A4 bölmesinden diğer bölmelere park edilirken fonksiyonun aldığı değerler Çizelge 5.16'de hesaplanmıştır.

Çizelge 5.16. Aracın kabul bölmesinden (A4) diğer bölmeler yerleştirilirken uygunluk fonksiyonunun aldığı değerler

Aracın Alındığı Bölme	Aracın Yerleştirildiği Bölme	Uygunluk Fonksiyonunun Aldığı Değer
A4	A1	0,183655
A4	A2	0,275482
A4	A3	0,550964
A4	A5	0,550964
A4	A6	0,275482
A4	A7	0,183655
A4	B1	0,130191
A4	B2	0,170474
A4	B3	0,246853
A4	B4	0,447227
A4	B5	0,246853
A4	B6	0,170474
A4	B7	0,130191

Fonksiyonun aldığı değerler dikkate alınırsa A4 bölmesine göre park edilme sırası yine A3, A5, B4, A2, A6, B3, B5, A1, A7, B2, B6, B1, B7 şeklinde olmaktadır.

5.5.5. Yapay zekâ

Yapay zekâ, insanoğlunun zekâ gerektiren davranışlarını taklit eden akıllı bilgisayar sistemlerini tasarlamak ve geliştirmekle uğraşan bilgisayar biliminin bir dalıdır. Uzman sistemler ise yapay zekâ metotlarından biridir.

Uzman sistem programları genel anlamda "Muhakeme Etme"; yani eldeki verilere göre en uygun davranışı belirleme prensibine göre çalışırlar. Genellikle bilgi tabanındaki kuralların muhakeme edilmesi iki yöntemle gerçekleştirilir.

- İleriye Doğru Zincirleme
- Geriye Doğru Zincirleme

İleriye doğru zincirleme tekniğinde problemin en başından başlanarak (IF cümlesinden) sonuç kısmına (THEN. . .) ulaşılır. Geriye doğru zincirlemede ise problem çözülürken kuralın en sonu olan sonuç (THEN. . .) cümlesi ile başlanır ve şart (IF. . .) cümleleri tatbik edilerek çözüm üretilir.

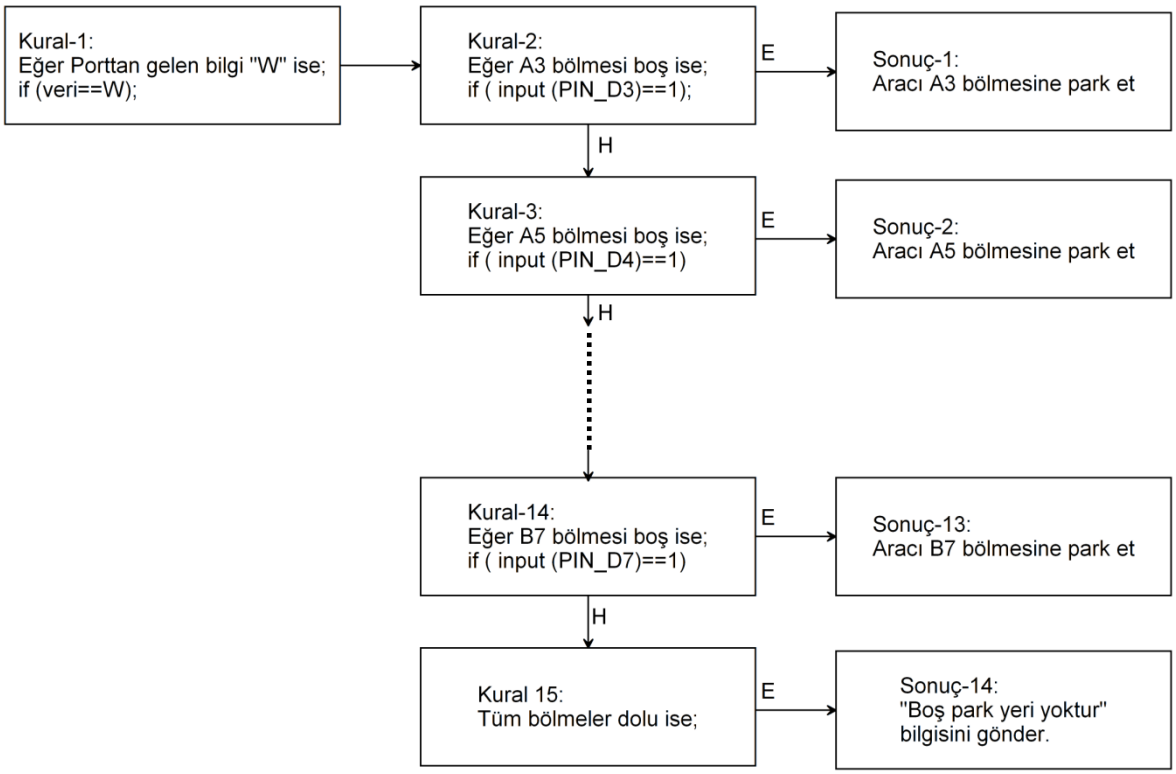
Uzman sistemler tekniğinin çalışmaya uygulanması

Bu tezde uygulaması gerçekleştirilen sistemin kontrolünü ve yönlendirmesini yapan mikrodenetleyiciye yüklenen program ile yapay zekâ tekniklerinden biri olan uzman sistemler tekniği kullanılmıştır. Gelen aracın park edilmesi işleminde en uygun bölmenin tespit edilmesi için bu teknikten faydalanılmıştır. Program "IF…THEN…ELSE" yapısına göre kural tabanlı bir sistem olarak çalışmaktadır. Kural tabanlı sistemler IF-THEN kurallarından, olaylardan ve bunların yorumlanmasından oluşur. Kural tabanlı sistemler sonuç çıkarım tekniklerinden bir tanesidir. Sonuç çıkarımı için, sistem elinde bulunan kuralları kullanarak ne yapacağına karar verir. Şekil 5.34'de bu yapı ile oluşturulan kural tabanının örnek bir parçası verilmiştir.

```
if (veri=='W')
   {
   if(input(PIN_D3)==1 )
      {
         a4den_a3e_koy ();
         veri=' ';
         putc ('C');
      }
   else if(input(PIN_D4)==1 )
      { .
   .
   .
```

Şekil 5.34. Kural tabanına ilişkin örnek kod satırları

Sonuca ulaşmak için ileriye doğru zincirleme metodunu kullanılmıştır. Yani problemin en başından başlanarak IF deyiminden sonraki şartlar sağlanıyorsa sonuç kısmına (THEN. . .) ulaşılmakta ve ne yapılacağına karar verilmektedir. Şekil 5.35'deki blok diyagramda ileriye doğru zincirleme metodunun yapısı görülmektedir.



Şekil 5.35. İleriye doğru zincirleme metodunun yapısı

Veri tabanına yüklenmiş 15 kural vardır. Birinci kural ön şarttır ve birinci kural sağlanmadan diğer kuralların sağlanıp sağlanmadığına bakılmaz. Eğer birinci kural sağlanıyorsa, sonuca ulaşmak için, sırasıyla diğer kurallar kontrol edilir. Gözden kaçırılmaması gereken nokta şu ki; birinci kuraldan sonra diğer kurallar sırası ile kontrol edilir. İkinci kural sağlanıyorsa sonuç-1'e ulaşılır. Eğer ikinci kural sağlanmıyor ise üçüncü kural kontrol edilir ve doğruysa sonuç-2'ye ulaşılır. İkinci kural da sağlanmıyor ise sırası ilde diğer kuralların sağlanıp sağlanmadığı kontrol edilerek diğer sonuçlardan birine ulaşılır.

Örnek üzerinde incelemek gerekirse; bilgisayar yazılımı, park etmek üzere bir aracın beklediğini seri port üzerinden PIC'e "W" bilgisini göndererek bildirir. Bu bilgi ile birinci kural sağlanmış olur. Daha sonra PIC mikrodenetleyicisi A3 bölmesinin boş olup olmadığını kontrol eder. Eğer boş ise ikinci kural da sağlanmış olur. Sonuç çıktısı olarak da araç A3 bölmesine park edilir. Eğer A3 bölmesi boş değil ise A5 bölmesi kontrol edilir ve boş ise sonuç çıktısı olarak araç A5 bölmesine park edilir. Eğer A5 bölmesi de boş değil ise sırası ile diğer bölmeler kontrol edilerek sonuç çıktısı elde edilmeye çalışılır. Bölmelerin tamamı dolu ise sonuç çıktısı "Boş Park Yeri Yoktur" bilgisidir.

Bilindiği üzere uzman sistemler; bir uzmanın bilgi, deneyim ve tecrübelerinin sisteme aktarılarak problemler karşısında veya giriş verilerine göre uzmanın düşünme ve çözüm geliştirme yeteneğinin taklit edilmesidir. Yapılan çalışmada veriler değerlendirilerek en uygun park yeri sıralaması belirlenmiş (Bkz. 5.5.4. Park edilecek en uygun bölmenin tespit edilmesi) ve bu bilgi yazılımda veri tabanına kaydedilmiştir. Kural tabanına göre de bölmelerin durumu bu sıralamaya göre kontrol edilerek (muhakeme ünitesi) en uygun park bölmesi belirlenmektedir. Belirlene park bölmesi uzman sistemin sonuç çıktısıdır.

5.6. Bilgisayar Yazılımı

Bilgisayar yazılımı, PIC mikrodenetleyicisi ile iletişim kurabilen bir kullanıcı arayüzüdür. C# programı ile yazılmış olup Windows tabanlıdır. Arayüz yazılımının çalışması için dotNet FrameWork programının 2.0 sürümü veya üstü bilgisayarda kurulu olmalıdır. Bu program oluşturulurken, sık kullanılması, kaynakların çokluğu, Windows ve Microsoft Access gibi veri tabanlarıyla uyumluluğundan dolayı C# seçilmiş ve uygulama gerçekleştirilmiştir.

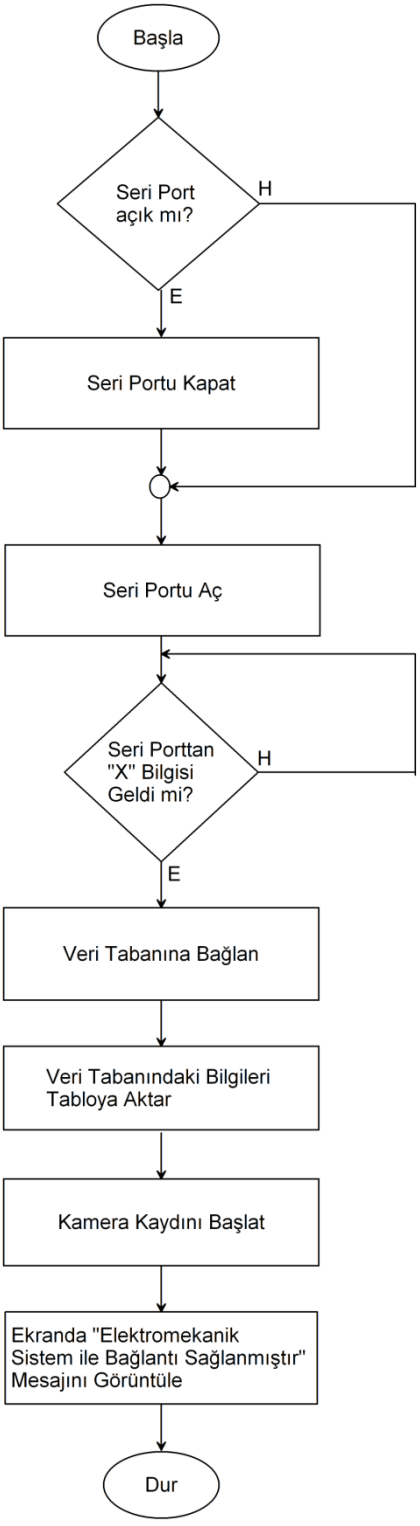
Arayüz programı, park etmek için gelen araçların bir WebCam yardımı ile fotoğrafını çekip, bu fotoğraftan plaka tanımlaması yapmaktadır. Plaka tanımlama işlemi için OCR (Optical Character Recognition – Optik Karakter Tanımlama) metodu kullanılmaktadır. Plaka tanımlama işlemi tamamlanıp araç park edildikten sonra PIC'den gelen park yeri bilgisini alır ve veri tabanına araç plakasını, park yeri bilgisini ve giriş zamanının kaydeder. Aracın çıkışı yapılmak istendiğinde ise girilen park yeri bilgisini PIC'e göndererek aracın çıkış işlemini başlatır ve çıkış zamanına göre park ücretini hesaplar. Ayrıca aynı aracın otoparkı kaç kez kullandığı bilgisini de veri tabanından kontrol ederek ekrana yansıtmaktadır.

PIC ile bilgisayar yazılımı arasındaki iletişim RS-232 protokolü ile yapılmaktadır. Elektromekanik sistem ile yazılım, seri port ile 9600 baud veri hızı ve 8 veri biti ile haberleşmektedir. Bu ayarlar hem PIC'te hem de yazılımda yapılmıştır.

5.6.1. Programın açılması

Bilgisayar yazılımı açıldığında ekrana Resim 5.27'de görülen arayüz gelir. Arayüzde kamera görüntüsünün anlık olarak izlenebildiği bir alan, çekilen fotoğraftan kesilen plaka bölgesinin görüntülendiği bir alan, plaka okuma işlemi neticesinde programın tanımladığı plaka bilgisinin görüntülendiği bir metin kutusu, manuel olarak fotoğraf çekimine olanak sağlayan "Foto Çek" butonu, aracın park edilmesi işlemini başlatan "Giriş" butonu, park edilmiş bir aracı müşteriye teslim etme işlemini başlatan "Çıkış" butonu, park edilen araçlara ilişkin bilgilerin görüntülendiği bir tablo ve istenilmesi halinde tablodaki bilgileri silen "Kayıtları Sil" butonu yer almaktadır. Ayrıca sol tarafta, butonların yanında otopark bölmelerinin boş yada dolu olma durumunu bildiren kutucuklar vardır. İlgili kutucuk yeşil ise park bölmesi boş, kırmızı ise doludur.

Resim 5.27. Bilgisayar yazılımı arayüzü



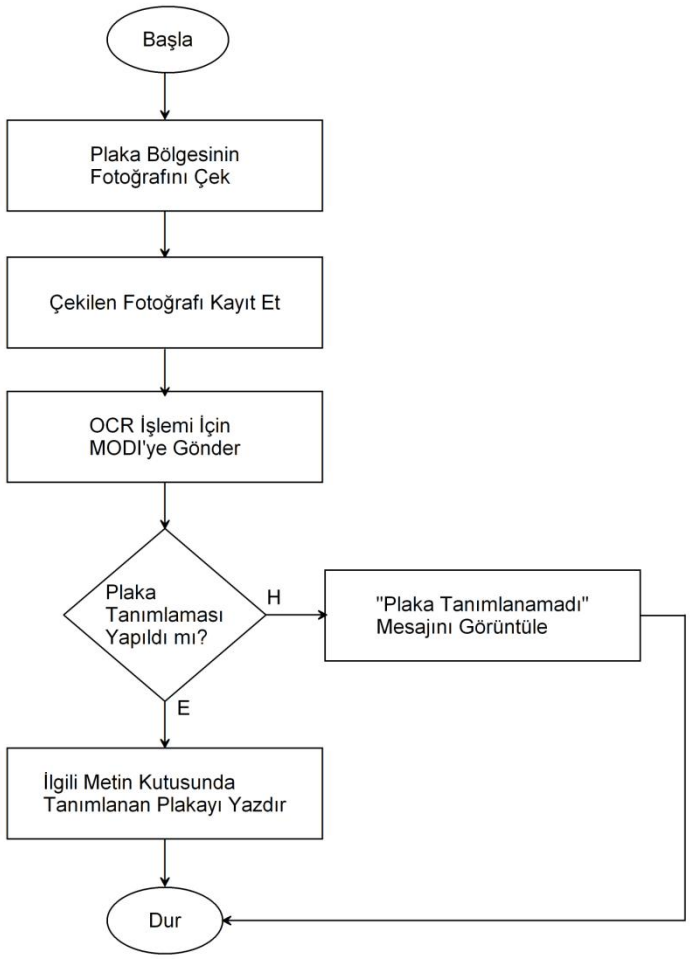
Şekil 5.36. Program açılışı akış şeması

Program ilk açıldığında ki işleyişi gösteren akış diyagramı Şekil 5.36'da görülmektedir. Öncelikli olarak seri portun açık olup olmadığı kontrol edilir. Seri port kapalı ise açılır. Eğer açıksa seri port kapatılır. Sonra tekrar açılır.

Açık olan seri portun kapatılıp tekrar açılmasındaki amaç veri güvenliğini sağlamak ve oluşabilecek muhtemel bir hatanın önüne geçmektir. Seri portu açma işlemi tamamlandıktan sonra elektromekanik sistemden açılış kodunun (“X” bilgisi) gelmesini bekler. Bu kod gelmeden program açılmaz ve ekrana hiçbir şey yansımaz. Açılış kodu geldikten sonra veri tabanına bağlanır ve varsa önceki kayıtları alarak tabloya aktarır. Daha sonra kamera kaydı başlatılır. Kamerada ya da bağlantısında bir sorun yok ise kamera görüntüsü ilgili alanda anlık olarak görülür. Program ile elektromekanik sistem arasındaki iletişim bağlantısının sağlıklı bir şekilde kurulduğunu belirten “Elektromekanik Sistem ile Bağlantı Sağlanmıştır” mesajı ekranda görülür (Resim 5.28). Artık program açılmıştır ve işlem yapmaya hazırdır.

Resim 5.28. İlk açılış ekranı

5.6.2. Park etmek için gelen aracın fotoğrafının çekilmesi



Şekil 5.37. Otomatik fotoğraf çekimi akış şeması

Şekil 5.37'de görülen akış diyagramında park etmek üzere, otoparkta araç kabul ve teslim bölmesi (slot) olarak belirlenen A4 bölmesine araç girişi yapıldığında gerçekleşen olaylar görülmektedir. Bu işlemler program satırlarında "okuma fonksiyonu" içerisinde gerçekleşmektedir. Araç A4 bölmesine giriş yaptığında sensör aracı algılar ve PIC mikrodenetleyicisi programa fotoğraf çekmesi için gerekli bilgiyi ("Q" bilgisi) gönderir. Akış diyagramı bu bilginin geldiği kabul edilerek hazırlanmıştır. Bilgiyi alan program fotoğraf çekme işlemini yapar ve plaka bölgesinin olduğu kısmı keserek kaydeder. Kaydedilen fotoğraf tanımlama işleminin yapılabilmesi için MODI'ye gönderilir. Resimden plaka tanımlaması başarı ile yapılabilirse ilgili metin kutusunda tanımlanan plaka görüntülenir. Resim 5.29'da bu ekran görüntüsü görülmektedir.

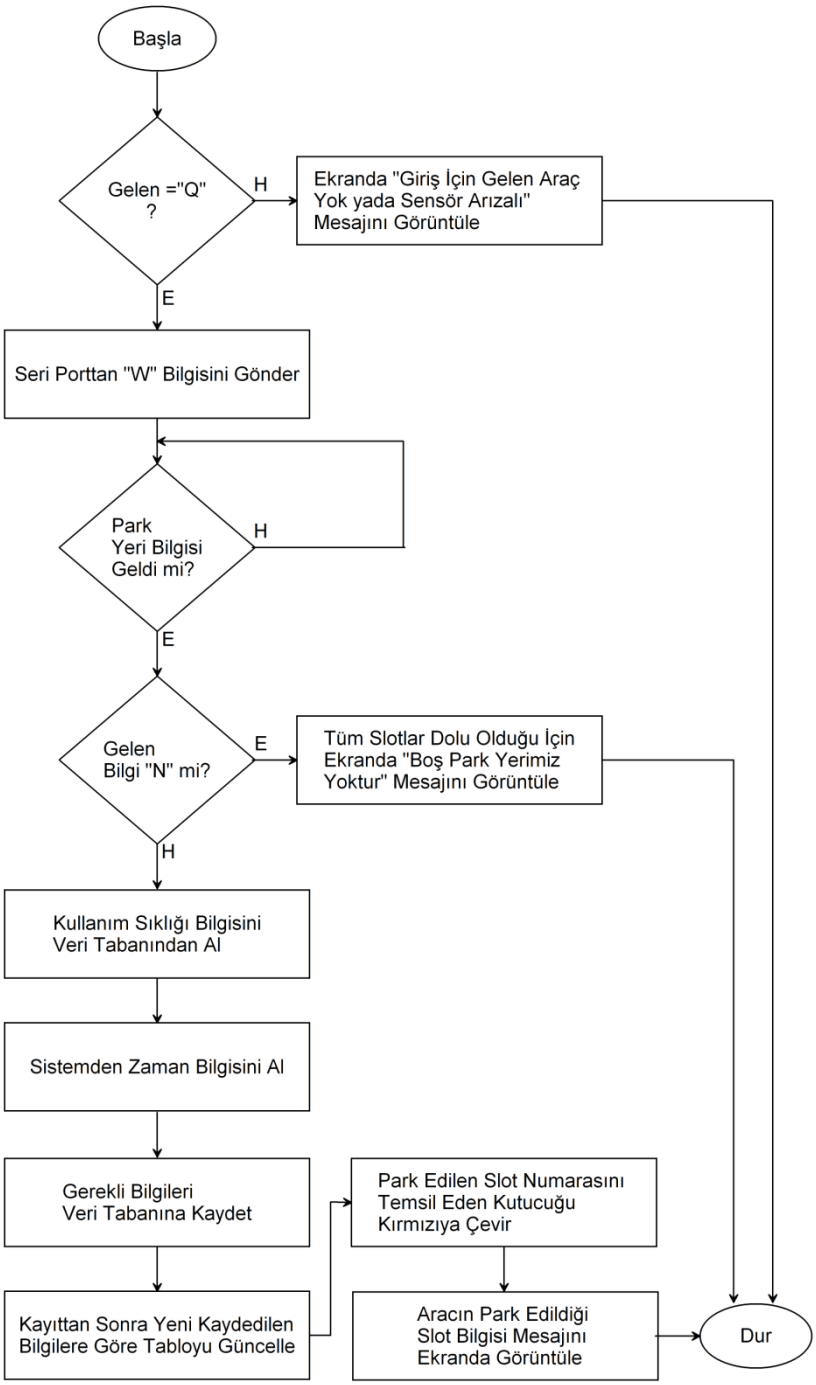
Resim 5.29. Plakanın tanımlanması

Tanımlanan plakada eksiklikler yada hatalar var ise kullanıcın klavyeden bunu düzeltmesi mümkündür. Aksi halde hatalı plaka bilgisi ile işlem yapılacaktır. Plaka tanımlama işlemi hiç yapılamazsa ekranda “Plaka Tanımlanamadı” mesajı görülür (Resim 5.30.). Bu durumda yeniden plaka tanımlama işlemi yapılabilmesi için aracın çıkıp tekrar girmesi gerekir yada “Foto Çek” butonuna basılması gerekir. “Foto Çek” butonu bu ve benzeri durumlarda plaka tanımlama işleminin yapılabilmesi için kullanıcıya sağlanmış bir kolaylıktır.

Resim 5.30. Plakanın tanımlanamaması durumu

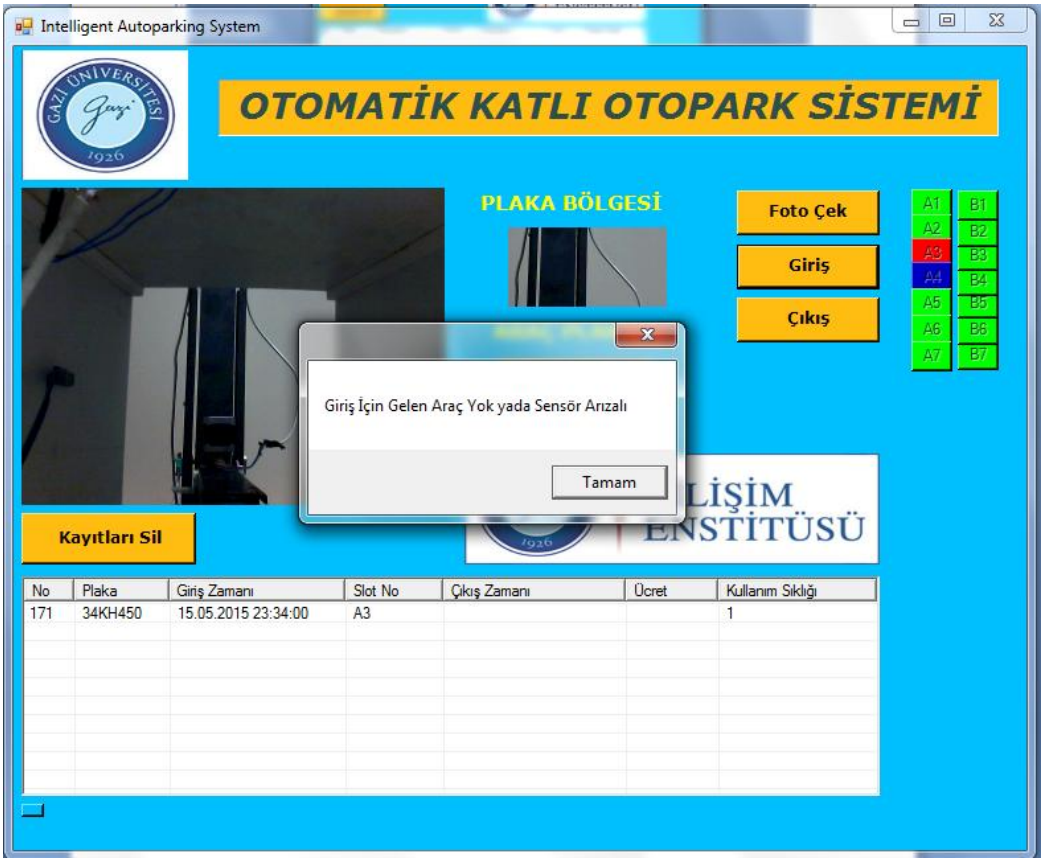
5.6.3. Araç girişinin yapılması

Daha önce kabul bölmesine gelen ve fotoğrafı çekilip plaka tanımlaması yapılan aracın park edilme işlemlerinin başlatılabilmesi için “Giriş” butonuna basılması gerekir. Şekil 5.38’de ki akış diyagramı bu aşamaları göstermektedir.



Şekil 5.38. Araç girişi akış şeması

Giriş butonuna basıldığında programda bulunan “Gelen” değişkeninin içeriğinin kontrol eder. Bu değişkenin içeriği “Q” olmalıdır. Bunun kontrol edilmesindeki amaç fotoğraf çekimi ve plaka tanımlaması yapılmadan “Giriş” butonuna basılırsa hem hatalı ve yanlış bir işlem yapılmasının önüne geçmek hem de programın kilitlenmesini engellemektir. Eğer “Gelen” değişkeninin içeriği “Q” değil ise ekranda “Giriş için Gelen Araç Yok yada Sensör Arızalı” mesajı görülür (Resim 5.31).



Resim 5.31. Fotoğraf çekilmeden giriş butonuna basılınca alınan hata mesajı

Eğer değişkenin içeriği “Q” ise seri porttan PIC’e “W” bilgisi gönderilir. Bu bilgi PIC’e park etmek üzere gelen bir araç olduğunu ve en uygun yere park etmesi gerektiğini bildirir. PIC bu bilgiyi alır almaz park bölmelerini tarar ve boş yer varsa en uygun yere park eder. Program, park etme işlemi tamamlanıp PIC’den park edilen bölmeye ilişkin bilgi gelene kadar döngüde bekler. Park etme işlemi tamamlandıktan sonra PIC park edilen bölmeye göre “A”…”M” arası bir bilgi gönderir. Hangi bilginin hangi bölmeye ait olduğu Çizelge 5.12’de belirtilmişti. Eğer boş park yeri yoksa PIC “N” bilgisi gönderir. Her durumda PIC bir bilgi göndermektedir. Program gelen bilgiyi değerlendirir. Eğer “N” bilgisi gelmiş ise ekranda “Boş Park Yerimiz Yoktur” mesajı görülür (Resim 5.32).

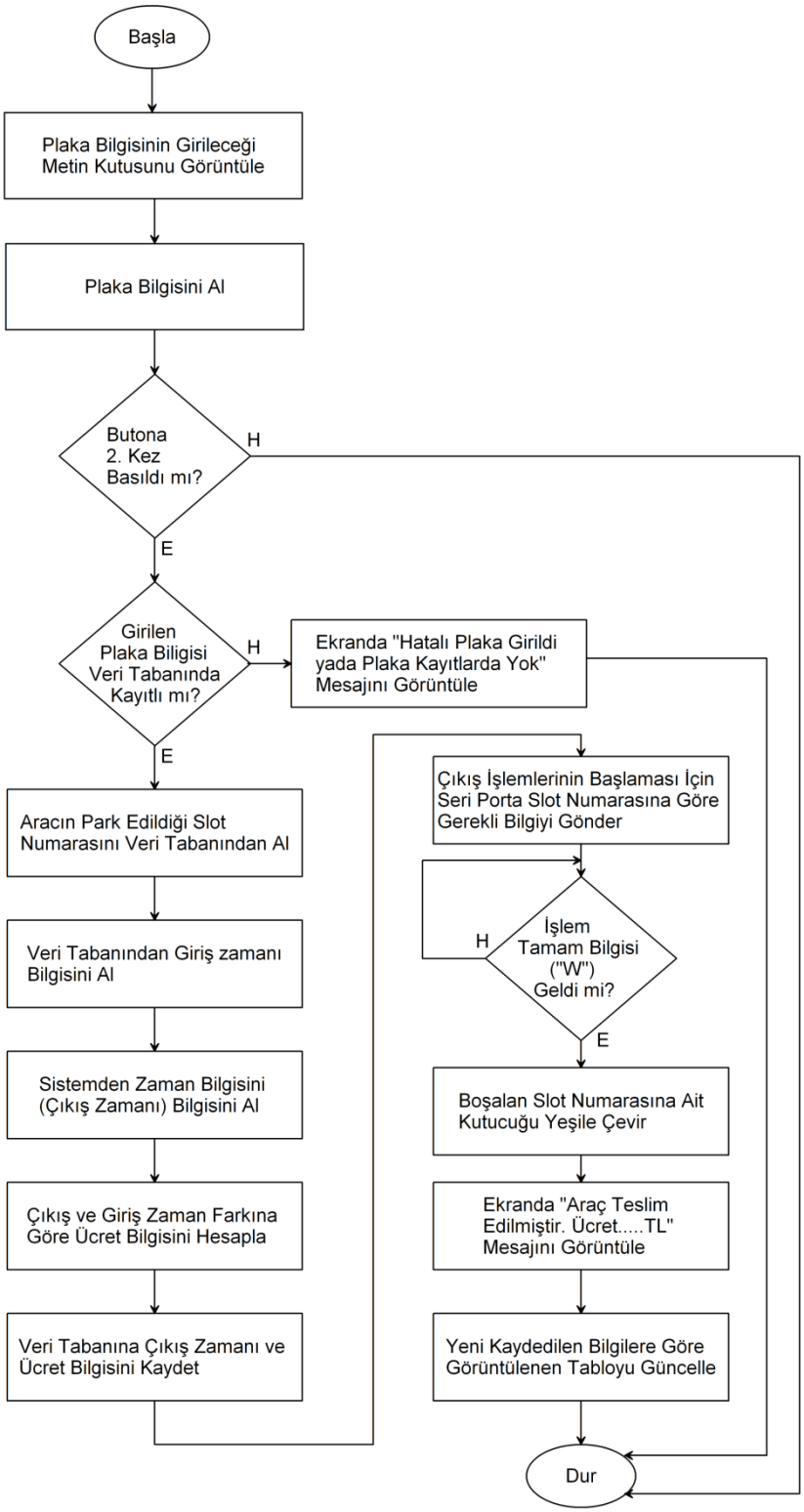
Resim 5.32. Boş park yeri olmaması durumunda alınan mesaj

Aracın park edildiği bölmeye ilişkin “A”…”M” arası bir bilgi gelmiş ise araç park edilmiştir ve plakaya göre kullanım sıklığı bilgisi veri tabanından alınır. İşletim sisteminden zaman bilgisi alınarak gerekli veriler (plaka, slot no, giriş zamanı ve kullanım sıklığı) veri tabanına kaydedilir. Kayıttan sonra yeni kaydedilen bilgilerin tabloda görünmesi için tablo güncellenir. Daha önce boş olduğu için yeşil olan ilgili slot numarasına ait kutucuğun rengi kırmızıya döner. Ayrıca ekranda aracın park edildiği bölme/slot bilgisi ekranda mesaj olarak görüntülenir (Resim 5.33).

Resim 5.33. Park etme işlemi tamamlandıktan sonra ekran görüntüsü

5.6.4. Araç çıkışının yapılması

Daha önce park edilmiş bir aracın teslim edilmesi için çıkış işlemlerinin yapılması gerekir. Şekil 5.39'da görülen akış diyagramı "Çıkış" butonuna basıldıktan sonra gerçekleşen olayları göstermektedir.

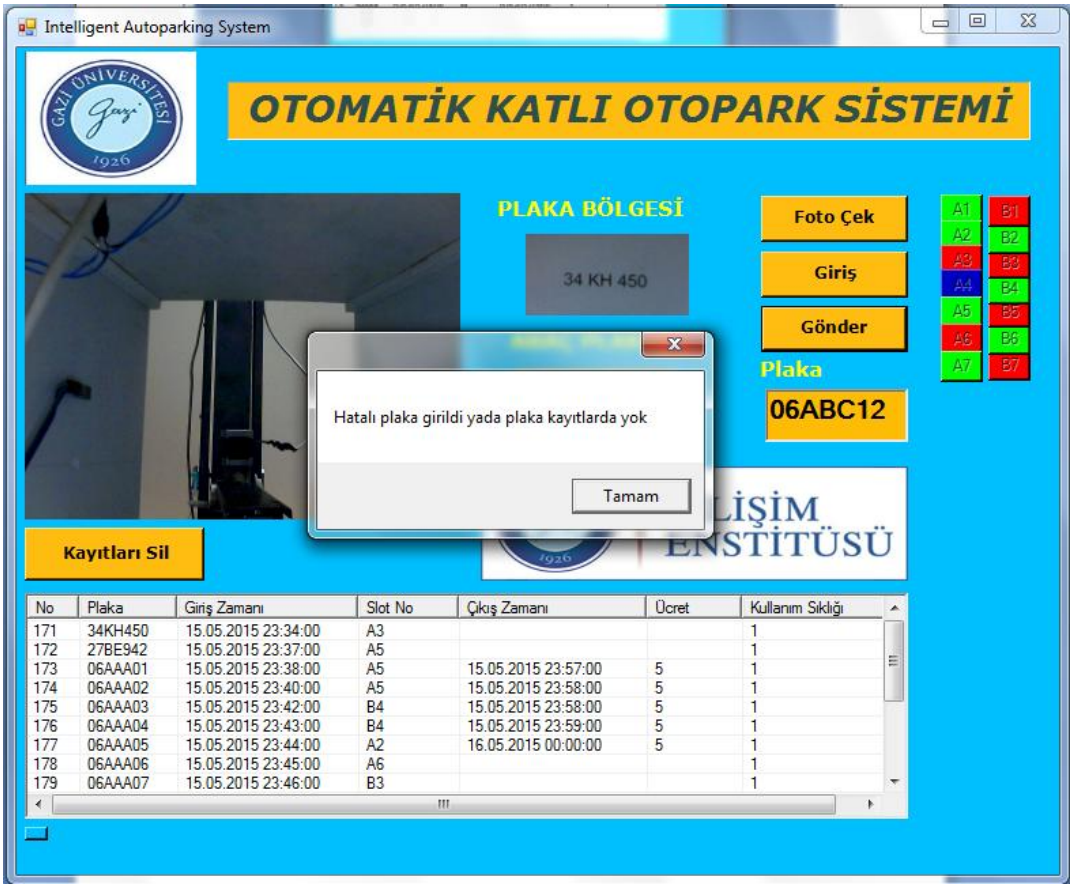


Şekil 5.39. Araç çıkışı akış şeması

“Çıkış” butonuna basıldığında normalde ekranda olmayan ve daha önce park edilen araca ilişkin plaka bilgisinin girilmesini sağlayan boş bir metin kutusu hemen butonun altında görülür. Buton da bu işlemden sonra isim değiştirir ve “Gönder” adını alır. Bu durum Resim 5.34’de görülmektedir.

Resim 5.34. Çıkış butonuna basıldıktan sonraki ekran görüntüsü

Açılan metin kutusuna teslim edilecek araca ilişkin plaka bilgisi girilir ve program isim değiştiren aynı butona ikinci kez basılmasını bekler. “Gönder” butonuna basıldığında girilen plaka bilgisi veri tabanındaki bilgiler ile kontrol edilir. Eğer geçerli bir bilgi değil ise veya girilen plaka bilgisine ilişkin daha önce oluşturulmuş bir kayıt yok ise ekranda “Hatalı plaka girildi yada plaka kayıtlarda yok” mesajı görüntülenir (Resim 5.35). Bu durumda Giriş butonuna tekrar basılıp plaka bilgisinin doğru bir şekilde tekrar girilmesi gerekir. Ya da başka bir işleme geçilebilir.



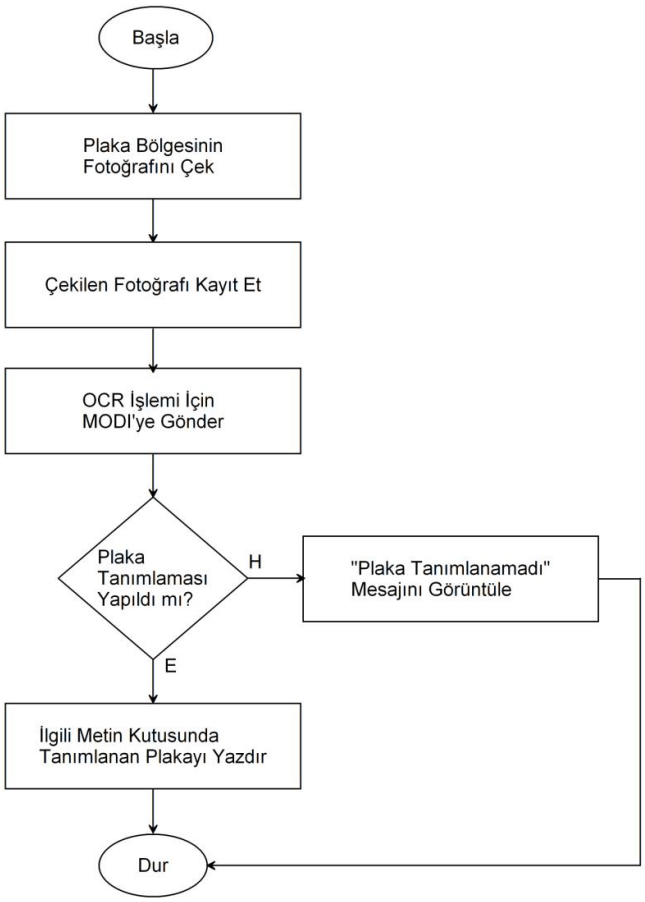
Resim 5.35. Slot numarasının hatalı girilmesi durumu

Plaka bilgisi doğru ise girilen plaka bilgisine göre veri tabanından aracın park edildiği slot numarası ve giriş zamanı bilgisi alınır, sistemden o anki zaman bilgisi (çıkış zamanı) alınır. Çıkış ile giriş arasındaki zaman farkına göre ücret bilgisi hesaplanır. Daha sonra çıkış zamanı bilgisi ve ücret bilgisi veri tabanına kaydedilir. Seri porttan PIC'e slot numarasına göre gerekli bilgi gönderilerek çıkış işlemlerinin başlatılması sağlanır. Çıkış işlemi bitene kadar program döngüde bekler. PIC çıkış işlemini tamamladıktan sonra "W" bilgisini gönderir. Bu bilgiyi alan program teslim işleminin bittiğini anlar ve döngüden çıkarak önce boşalan slot numarasına ait kutucuğu yeşil renge çevirir. Sonrasında ise ekranda "Araç Teslim Edilmiştir. Ücret….TL" mesajını görüntüler. Kaydedilen bilgiler göre tablo yeniden güncellenir. Resim 5.36'da bu ekrana ait görüntü incelenebilir.

Resim 5.36. Araç teslim edildikten sonraki ekran görüntüsü

5.6.5. Manuel fotoğraf çekme işlemi

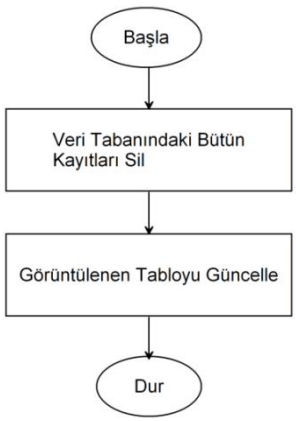
Plakanın hatalı veya eksik tanımlanması durumunda işlemin tekrarlanması için fotoğraf çekme işleminin yeniden yapılması gerekir. Bunun için, daha önce bahsedildiği üzere ya araç çıkıp tekrar giriş yapmalı ya da manuel olarak fotoğraf çekilmelidir. “Foto Çek” butonuna basılırsa fotoğraf çekme işlemi yapılır ve Şekil 5.40’da görülen akış diyagramından da anlaşılacağı üzere tanımlama işlemleri yapılır. Bu işlemler daha önce anlatılanlar ile aynıdır.



Şekil 5.40. Manuel fotoğraf çekimi akış şeması

5.6.6. Kayıtların silinmesi

Tabloda görüntülenen kayıtlar silinmek istenirse “Kayıtları Sil” butonuna basılmalıdır. Bu butona basıldığında veri tabanındaki bütün kayıtlar silinir ve tablo güncellenerek boş tablo görüntülenir.



Şekil 5.41. Kayıtların silinmesi akış şeması

6. SONUÇ ve ÖNERİLER

İçinde bulunduğumuz zaman diliminde hız, zaman, prestij, konfor ve teknoloji insanlar için çok önemli duruma gelmiştir. Bu bağlamda klasik otoparklar incelendiğinde, özellikle kalabalık şehirlerde artık ihtiyaçlara cevap veremediği ortaya çıkmaktadır.

Bu tezde, park etmek üzere sürücüsü tarafından teslim edilen bir aracı taşıyıcı bir robot mekanizması ile alıp en uygun yere park eden ve bilgisayar yazılımı ile aracın plaka tanımlamasını yapan bir sistemin prototip uygulaması gerçekleştirilmiştir. Park etme esnasında aracın taşınabilmesi için Japon Endüstriyel Robot Birliği'ne (JIRA) göre 5. sınıf, açık döngü kontrol sistemli, silindirik koordinatlı (R2P), elektrik tahrik sistemli, programlanabilme özelliğine sahip bir robot sistemi tasarlanmıştır. Tasarlanan otopark sistemi "dairesel otomatik otopark" sınıfına girmektedir.

Yapılan çalışmada, park etmek için aracın gelip gelmediği ve park bölmelerinin boş yada dolu olma durumu kızılötesi yakınlık sensörleri ile tespit edilmektedir. Robot sisteminin eksensel hareketleri için adım motorlar kullanılmıştır. Robot sisteminin hareket sınırlarını belirlemek için gerekli yerlerde mekanik anahtarlar ve manyetik röleler kullanılmıştır. Elektronik devrenin kontrolü ve işleyişi PIC 16F877A mikrodenetleyicisi tarafından gerçekleştirilmektedir. Plaka tanımlanmasında kullanılan yazılım ile elektronik devre RS-232 protokolü ile haberleşmektedirler. Araçların fotoğrafları bir kamera yardımı ile çekilip bilgisayara aktarılmaktadır.

PIC mikrodenetleyicisine yüklenen program ile aracın park edileceği bölme en kısa yoldan ve en kısa sürede park edilecek şekilde tespit edilmektedir. Böylece sisteme hem zaman hem de enerji tasarrufu özelliği kazandırılmıştır. En uygun park yerinin tespitinde yapay zeka metodlarından biri olan; ileriye doğru zincirleme yöntemi ile sonuca ulaşan, kural tabanlı uzman sistemler metodu kullanılmıştır. Kullanılan kızılötesi yakınlık sensörleri park bölmelerinin boş ya da dolu olma durumlarını kesin olarak bildirdiklerinden dolu olan bölmeye başka bir aracın girmesi mümkün olmamakta ve araçların güvenliği sağlanmaktadır.

Araç park edilirken veya teslim edilirken bilgisayar yazılımı PIC mikrodenetleyicisinden “işlem tamam” bilgisini bekler ve bu bilgi geldikten sonra sıradaki işlemi gerçekleştirir. Bu durum sistem güvenliğini ve kararlılığını artırmaktadır.

Uygulaması geçekleştirilen otomatik otopark sisteminin yukarıda bahsedilen özelliklerinin dışında bir de plaka tanımlama özelliği vardır. Bu özellik sistemin daha işlevsel ve kullanışlı olmasını sağlamaktadır. Plaka tanımlama işleminde fotoğraf çekimi ve plakanın tanımlanması tamamen otomatik olarak yapılmakta, kullanıcının herhangi bir müdahalesine gerek kalmamaktadır. Park edilen bir araç çıkış yapacağında ise park süresine göre ücret otomatik olarak hesaplanıp ekrana yansıtılmaktadır. Eğer plaka tanımlanması esnasında program bazı karakterleri yanlış ya da eksik belirlerse kullanıcının bunu düzeltmesi mümkündür. Plaka tanıma programı Windows tabanlı olup her bilgisayarda çalıştırılabilir.

Sistem geliştirlmeye ve üzerine bileşenler eklenmeye uygun olarak tasarlanmıştır. Bilgisayar programında kullanıcı yetkilendirilmesi yapılarak her kullanıcıya yetki verilebilir. Böylece hem izinsiz kişilerin sisteme erişimi engellenecek hem de her kullanıcının yaptığı işlemler kayıt altında tutulacaktır.

Abone özelliği programa eklenerek otoparkı her gün kullanan müşteriler için yer ayarlaması yapılabilir. Aboneler için indirim uygulanıp müşteri sayısı artırılabilir.

Bilgisayar programına rapor alma özelliği eklenerek gün sonunda veya seçilen zaman aralıklarına göre kullanım detayları alınabilir. Bu detaylar kullanım sıklığı, saatlere göre yoğunluk, ücret, seçilen bir plakaya göre kullanım istatistikleri vb. olabilir. Bu bilgilere göre sistemin çalışması programlanabilir veya geliştirilebilir.

Bilgisayar yazılımına ağ üzerinden erişim özelliği kazandırılarak yöneticilerin istedikleri zaman kendi bilgisayarlarından rapor alabilmeleri ve izleyebilmeleri sağlanabilir.

Sistemde kullanılan adım motorların dönüş açıları ile ilgili herhangi bir geribildirim yapılmamaktadır. Adım motorların millerine bağlanacak enkoderler geribildirim

yapılabilir veya adım motorlar yerine servo motorlar kullanılabilir. Böylece sistemin kararlılığı ve güvenilirliği artacak, kaza ihtimali azalacaktır.

Sisteme eklenecek çeşitli sensörler ile araçların renk, ağırlık vb. özellikleri tespit edilebilir. Bu bilgiler çeşitli amaçlar için kullanılabilir. Örneğin ağırlık bilgisine göre sistemin harcayacağı enerji dikkate alınarak park yeri tespiti yapılabilir.

Park bölmelerine yangın algılama sensörleri yerleştirilerek olası bir yangın durumunda o bölgedeki araçların tahliye edilmesi ve yangına müdahale edilmesi sağlanabilir.

Prototip niteliğinde uygulaması gerçekleştirilen otomatik otopark sisteminin gerçek uygulamasında ortaya çıkacak avantajları ise aşağıda sıralanmıştır.

- Park yeri arama zahmetini ortadan kaldırarak sürücüye zaman ve yakıt tasarrufu sağlayacaktır.
- Park etmek üzere gelen aracın motoru durdurulacağından araba egzozlarından doğaya salınan zararlı gazlar azalacaktır.
- Sürücü otoparka girmediğinden zararlı toz ve gazlara maruz kalmayacaktır. Sistem bu yönü ile insan sağlığını korumaya yardımcı olacaktır.
- Araçların ve insanların saldırılara maruz kalarak zarar görmeleri önlenebilir.
- Klasik otoparklarda zorunlu olarak bulunan rampalar, araç manevra alanları, yaya merdivenleri/asansörleri, araçların giriş-çıkış esnasında kullanacağı yollar otomatik otoparklarda bulunmayacağından ve araçlar arasındaki mesafeler daha az olacağından aynı kapasitede bir otomatik otopark, klasik otoparklara göre daha küçük arazilere yapılabilir.

Yapılan çalışmanın diğer çalışmalara (Bkz. 2. bölüm) göre; kapasitesinin fazla olması, her kat için ayrı elektromekanik sistem gerektirmemesi, her bir işlem için sadece tek bir aracı taşıyacak kadar güce ihtiyaç duyması, park etme ve teslim etme işlemlerinin basit ve sade olması, geri dönüşüme müsait olması gibi üstünlükleri vardır. Ayrıca plaka tanıma özelliği de diğer çalışmalara göre önemli bir avantajıdır.

Prototip niteliğindeki bu çalışmanın gerçek anlamda kullanılması durumunda aşağıdaki hususlar göz önüne alınmalıdır.

Yapılan çalışma prototip niteliğinde olduğu için sadece bir taşıyıcı robot tasarlanmıştır. Gerçek bir uygulamada ise ihtiyaca göre birden fazla taşıyıcı sistem kullanılmalıdır.

Prototip uygulamada oyuncak araçlar kullanıldığı için araçlar altına giren bir palet ile kaldırılarak araç tekerleklerinin yer ile teması kesilmekte ve sonra taşınmaktadır. Bu yöntem gerçek uygulamalarda araçlara zarar verebilir. Bu yöntem yerine araçları tekerleklerinden kavrayabilecek veya araca zarar vermeyecek başka bir yöntem kullanılmalıdır.

Yapılan uygulamada aracın teslim alınması ve teslim edilmesi aynı bölmeden yapılmaktadır. Gerçek bir uygulamada bu yöntem uygun olmayıp uzun kuyruklara ve bekleme sürelerine neden olabilir. Çözüm olarak araçların kabulü ve teslimi ayrı yerlerden yapılmalıdır.

Plaka tanıma programı plakaların aynı yerde olduğu ve herhangi bir bozucu işaret (vida, ülke kodu vb.) olmadığı kabul edilerek hazırlanmıştır. Gerçek uygulamada çok daha profesyonel bir program kullanılmalıdır.

Araçları görüntülemek için basit bir web kamera kullanılmıştır. Gerçek uygulamada çok daha profesyonel, çözünürlüğü yüksek, ortamdaki ışık değişimlerini tolere edebilen kamera veya kameralar kullanılmalıdır.

Prototip uygulamada araçların aynı tip ve boyutta olduğu kabul edilmiştir. Gerçek uygulamada araçların boyutlarına ağırlıklarına ve tiplerine göre farklı park yerleri tasarlanıp yer kazancı sağlanabilir ve farklı araç türlerine hizmet verilebilir.

Yapılan çalışmada kaldırma, döndürme ve taşıma işlemleri için elektrik tahrikli motorlar kullanılmıştır. Gerçek uygulamada güç, verimlilik, bakım vb. teknik özellikler dikkate alınarak hidrolik ya da pnömatik sistemler tercih edilmelidir.

Gerçek uygulamada, taşıyıcı sistemin devre dışı kalma ihtimali göz önüne alınarak mutlaka yedek bir sistem bulundurulmalı ve en kısa sürede devreye alınabilecek şekilde tasarım yapılmalıdır. Çünkü hiç kimse aracını park yerinde bırakmak istemez.

Olası elektrik kesintilerinde sistemin çalışmasını etkilemeyecek şekilde on-line güç kaynakları ya da jeneratörler hazır olmalıdır.

Araçların zarar görmesini önlemek için hareket sınırlarını belirleyen veya hareketleri algılayan sensörler ile birlikte redundant sensörler kullanılmalıdır.

KAYNAKLAR

1. Yardım M. S. ve Ağrıklı M. (2005, 23-25 Mayıs). *Otomatik Otoparklar ve Türkiye'deki Otopark Probleminin Çözümü için Uygulama Potansiyeli.* TMMOB İnşaat Mühendisleri Odası İstanbul Şubesi 6. Ulaştırma Kongresinde sunuldu, İstanbul.

2. Şahin A. (2009). *İstanbul Merter Yer altı Otoparkı Örneği*, Yüksek Lisans Tezi, İstanbul Teknik Üniversitesi Fen Bilimleri Enstitüsü, İstanbul, 5,6,11-13.

3. Abboud N.W. (1994). *Automation of Parking Industry : A Strategic and Managerial Overwiev*, M. Sc. Thesis, Mussachussetts Institute of Technology, ABD, 17-19, 36, 43, 44, 47, 48.

4. Abu Hassan M. F. (2009). *Development of Automatic Parking System,* Degree of Bachelor, Universiti Teknikal Malaysia Melaka, Malezya, 2.

5. İnternet: *Multi Park.* URL:http://www.webcitation.org/query?url=http%3A%2F%2Fkatopark.com%2Fturkish%2Fautomated.htm%23%23%23+&date=2015-04-13, Son Erişim Tarihi: 25.09.2012.

6. İnternet: *Family Parking (DFP).* URL:http://www.webcitation.org/query?url=http%3A%2F%2Fwww.dyps.co.kr%2Fhtml_en%2Fproducts04.php&date=2015-04-13, Son Erişim Tarihi: 27.09.2012.

7. İnternet: *Combi Parker 555.* URL:http://www.webcitation.org/query?url=http%3A%2F%2Fwww.woehr.de%2Fen%2Fproduct%2Fitems%2Fcombiparker-555.html+&date=2015-04-13, Son Erişim Tarihi: 26.09.2012.

8. Bingöl O., Aydoğan T., Didin H.R., Yalçıner A.S. ve Duygulu K. (2010, Şubat). PLC Kontrollü Otomatik Katlı Otopark Sistemi, *SDU International Technologic Science*, 2 (1), 68.

9. Lo L. (2008). *Automated Parking System*, Degree of Bachelor, Universiti Teknikal Malaysia Melaka, Malezya, 14,15.

10. Sunitha K.A., Prema K., Deepthi G.S., Belinda E.J.E. and Kumar N.S. (2010, November), *Fuzzy Based Automatic Multi-Level Vehicle Parking Using Lab View*, Paper presented at the Frontiers in Automobile and Mechanical Engineering (FAME), IEEE International Conference, Chennai.

11. Mathijssen A. and Pretorius A.J. (2006). Verified Design of an Autoomated Parking Garage. *Proceedings of the 11th International Workshop, FMICS 2006 and 5th International Workshop*. Berlinn: Springer-Verlag, 165-180, 165-166.

12. Reza M.O., Ismail M.F., Rokoni A.A. and Sarkar M.A.R. (2012, April). Smart Parking System with Image Processing Facility, *International Journal of Intelligent Systems and Applications (IJISA),* 4 (3), 41-47, 41.

13. Milli Eğitim Bakanlığı (2007). *Elektrikli Ev Aletlerinde DC Motorlar.* Ankara: Milli Eğitim Bakanlığı, 3.

14. Hancı O. (2007). *Servo Motorlar ve Örnek Bir Uygulama Tasarımı*, Yüksek Lisans Tezi, Marmara Üniversitesi Fen Bilimleri Enstitüsü, İstanbul, 4, 12, 42, 65.

15. Uygun D. (2006). *Hibrid Adım Motorun Sayısal Kontrolü*, Yüksek Lisans Tezi, Marmara Üniversitesi Fen Bilimleri Enstitüsü, İstanbul, 5, 7, 8, 15, 21, 29, 30, 37-39.

16. Çelikel R. (2005). *Pic Mikrodenetleyici Yardımı ile Adım Motorunun Konum Kotrolü*, Yüksek Lisans Tezi, Fırat Üniversitesi Fen Bilimleri Enstitüsü, Elazığ, 7, 9-12, 15, 16, 21-23.

17. Yılmaz U., Küçük B. (2009). *Akıllı Güneş Takipleyici Sistem (Smart Solar Tracker)*, Bitirme Tezi, Yıldız Teknik Üniversitesi Elektrik-Elektronik Fakültesi Elektrik Mühendisliği Bölümü, İstanbul, 12,13, 20, 21.

18. İnternet: *ZM-2H606 İki Faz Step Motor Sürücüsü.* URL:http://www.webcitation.org/query?url=http%3A%2F%2Fwww.robosan.com.tr%2Fpdfs%2FZM-2H606%2520TURKCE.pdf+&date=2015-04-13, Son Erişim Tarihi, 18.12.2014.

19. Bodson M., jeffrey S. S. and Stephen R.S. (Aralık 2013). *Spontaneous Speed Revelsals in Stepper Motors*. Paper presented at the 42nd IEEE Conference on Desicion and Control Maui, Hawaii, ABD, 3.

20. Childs P.R.N. (2014). *Mechanical Design Engineering Handbook* (1st Edition). London: Butterworth – Heinemann, 460-462.

21. Erdöl M. (2014). *Bilyalı Vidalı Mil Hareket Sisteminin Matematiksel Modellenmesi ve Titreşim Analizi*, Yüksek Lisans Tezi, Gebze İleri Teknoloji Enstitüsü Mühendislik ve Fen Bilimleri Enstitüsü, Gebze, 5-6.

22. Bertoline G.R., Wiebe N.W., Miller C.L., Nasman O.L. (1995). *Technical Graphics Communication* (1st Edition). Kanada : Irwin Publisher, 1078.

23. Çiçek S. (2006). *Renge Göre (Kırmızı, Mavi, Yeşil) Malzeme Taşıyan Robot Kolu Tasarımı ve Uygulaması*, Yüksek Lisans Tezi, Gazi Üniversitesi Fen Bilimleri Enstitüsü, Ankara, 1, 18, 19, 22-34, 41-44, 53-56, 72.

24. Setchell C. J. (1997). *Application of Computer Vision to Road-Traffic Monitorring*, PhD Thesis, University of Bristol Departman of electrical and Electronical Engineering, Bristol, 1-6.

25. Bakkaloğlu A. (2011). *Araç Plaka Tanıma Sistemi*, Yüksek Lisans Tezi, Selçuk Üniversitesi Fen Bilimleri Enstitüsü, Konya, 1, 16, 18, 19, 21, 23, 24, 27.

26. Akar F. (2009). *Şablon Eşleme Yöntemi ile Plaka Tanıma ve Değerlendirme Sistemi*, Doktora Tezi, Atatürk Üniversitesi Fen Bilimleri Enstitüsü, Erzurum, 21, 22, 27, 28, 45, 46.

27. Draghici S. (1997). A Neural Network Based Artifical Vision System for License Plate Recognation Systems, *International Journal of Neural Systems*, 8 (1), 113-126.

28. Gonzalez R.C., Wintz P. (1987). *Digital Image Processing* (2nd Edition). USA: Addison-Wesley Publishing Company, 118-136.

29. Otsu N. (1979). A Threshold Selection Method from Gray-Level Histogram, *IEEE Transactions on Systems, Man, and Cybernetics*, 9 (1), 62-66.

30. Boztoprak H. (2007). *Gerçek Zamanlı Taşıt Plaka Tanıma Sistemi,* Yüksek Lisans Tezi, Süleyman Demirel Üniversitesi Fen Bilimleri Enstitüsü, Isparta, 20-22, 30.

31. Yalım B. (2008). *Türk Sivil Plaka Standartları için Araç Plaka Tanıma Sistemi*, Yüksek Lisans Tezi, Gazi Üniversitesi Bilişim Enstitüsü, Ankara, 10, 28, 29, 40.

32. Musyev E. (2004). *Bilgisayar Destekli Karakter Tanıma Sistemi Tasarımı*, Yüksek Lisans Tezi, İstanbul Üniversitesi Fen Bilimleri Enstitüsü, İstanbul, 4.

33. Çiçek S. (2007). *CCS C ile PIC Programlama* (1. Baskı). İstanbul: Altaş Yayınları, 15-32.

34. Bingül Z., Küçük S. (2005). *Robot Teknolojisi* (1. Baskı). İstanbul: Birsen Yayınevi, 104 – 200.

35. Craig J. J. (1989). *Introduction to Robotics: Mechanic and Control* (2nd Edition). USA: Addison-Wesley Publishing Company.

36. İnternet: *GP2Y0D810Z0F.*
Web: http://www.sharp.co.jp/products/device/doc/opto/gp2y0d810z_e.pdf adresinden 20 Aralık 2014'de alınmıştır.

37. Chan C.W., Jin H., Cheung K.C. and Zhang H.Y. (2001). Fault Dedection of Systems With Redundant Sensors Using Constrained Kohonen Networks, *Automatica,* 37(10), 1671.

38. Cho J.J., Chen Y. and Ding y. (2005). *Redundancy Analysis of Linear Sensor Systems and Its Applications,* Paper presented at the Informs Annual Meeting, San Francisco.

39. Julius T.T. (1990). A Konowledge-Based System For Redundant and Multi Sensing In Intelligent Robots, *Nato ASI Series,* F-58, 3,6,7,17.

40. Axelson J. (2000). *Her Yönüyle Paralel Port* (Çev. Cihan Gerçek). İstanbul: ERA Bilgi Sistemleri Yayıncılık. (Eserin orjinali 1997'de yayımlandı), 1-35.

41. Axelson J. (2000). *Her Yönüyle Seri Port* (Çev. Cihan Gerçek). İstanbul: ERA Bilgi Sistemleri Yayıncılık. (Eserin orjinali 1999'da yayımlandı), 1-12.

42. Axelson J. (2002). *Her Yönüyle USB Port* (Çev. Cihan Gerçek). İstanbul: ERA Bilgi Sistemleri Yayıncılık. (Eserin Orjinali 1999'da yayımlandı).

43. Richard W.D.N., Ramasubramanian R. (1995). *Interfacing the IBM-PC to Equipment: The Art of Serial Communication* (1st Edition). UK: Cambridge University Press, , 3-10, 12-17.

44. Machacek J. and Drapela J. (2008). Control of Serial Port (RS-232) Communication in Labwiev, *2008 International Conference Modern Technique and Technologies (2008 MTT) IEEE*, 24 (1), 36-40.

EKLER

EK-1. Mikrodenetleyici yazılımı

```
#include <16f877a.h> // Kullanılan mikrodenetleyici tanıtılıyor
#fuses XT, NOWDT, NOPROTECT, NOBROWNOUT, NOLVP, NOPUT, NOWRT,
NODEBUG, NOCPD
//Mikrodenetleyicinin konfigürasyon özellikleri tanıtılıyor
#use delay(clock=4000000) // 4MHz osilatör frekansı
#use rs232(baud=9600,xmit=pin_c6,rcv=pin_c7,parity=N,stop=1)
// RS232 iletişimi için ayarlar tanıtılıyor
int16 i;  // 16 bitlik değişken
int veri; // 8 bitlik değişken
int1 t;   // 1 bitlik değişken

//*************************************************************************************
void home_position()              //Başlangıç pozisyonuna gelme fonksiyonu
  {
  while (input(pin_a2)==1)        //Alt switch açık ise kapanana kadar aşağı iner
    {
      output_low(pin_b0);         //Yukarı/Aşağı hareketi sağlayan motor için yön
      output_low(pin_b1);
      delay_us(750);
      output_high(pin_b1);
      delay_us(750);
    }
  output_high(pin_b2);            //Sağ/Sol hareketi sağlayan motor için yön
  do                              //RA3 0 olana kadar döngü devam eder
                                   //Yan switch kapanana kadar palet sağa döner
    {
      output_high(pin_b3);
      delay_us(750);
      output_low(pin_b3);
      delay_us(750);
```

EK-1. (devam) Mikrodenetleyici yazılımı

```
    }
  while (input(pin_a3)==1);
  output_low(pin_b2);            //Sağ/Sol hareketi sağlayan motor için yön
  for (i=0;i<=7822;i++)          // i <= 7822 olduğu sürece döngü devam eder.
                                 //Döngü bittiğinde palet platformun ortasına gelir.
    {
      output_high(pin_b3);
      delay_us(750);
      output_low(pin_b3);
      delay_us(750);
    }
  output_high(pin_b0);           // Yukarı/Aşağı hareketi sağlayan motor için yön
  for (i=0;i<=302;i++)           // i <= 302 olduğu sürece döngü devam eder
                                 // Döngü bittiğinde palet bir miktar yukarı çıkar
    {
      output_high(pin_b1);
      delay_us(750);
      output_low(pin_b1);
      delay_us(750);
    }
  }
//*****************************************************************************************
void kat_cik()                   // Bir kat yukarı çıkma fonksiyonu.
  {
  output_high(pin_b0);           // Yukarı/Aşağı hareketi sağlayan motor için yön
  for (i=0;i<=3555;i++)          // i <= 3555 olduğu sürece döngü devam eder
                                 // Bu döngü bittiğinde palet bir üst kata çıkar
    {
      output_high(pin_b1);
      delay_us(750);
      output_low(pin_b1);
      delay_us(750);
```

EK-1. (devam) Mikrodenetleyici yazılımı

```
    }
  }
//*************************************************************************************************
void kat_in()                   // Bir kat aşağı inme fonksiyonu.
  {
  output_low(pin_b0);           // Yukarı/Aşağı hareketi sağlayan motor için yön
  for (i=0;i<=3555;i++)         // i <= 3555 olduğu sürece döngü devam eder
                                // Bu döngü bittiğinde palet bir alt kata iner
    {
      output_high(pin_b1);
      delay_us(750);
      output_low(pin_b1);
      delay_us(750);
    }
  }
//*************************************************************************************************
void ileri()                    //Paletin ileri hareket etmesini sağlayan fonksiyon
  {
  output_high(pin_b4);          // İleri/Geri hareketi sağlayan motor için yön
  do                            // RA0 0 olana kadar döngü devam eder
                                // Uçtaki röle kapanana kadar ileri gider
    {
      output_high(pin_b5);
      delay_us(750);
      output_low(pin_b5);
      delay_us(750);
    }
  while (input(pin_a0)==1);
  }
//*************************************************************************************************
void geri()                     // Paletin geri hareket etmesini sağlayan fonk.
  {
```

EK-1. (devam) Mikrodenetleyici yazılımı

```
  output_low(pin_b4);             // İleri/Geri hareketi sağlayan motor için yön
  do                              // RA1 0 olana kadar döngü devam eder
                                  // Gerideki reed röle kapanana kadar geri gelir
    {
      output_high(pin_b5);
      delay_us(750);
      output_low(pin_b5);
      delay_us(750);
    }
  while (input(pin_a1)==1);
  }
//*****************************************************************************************
void kaldir()                     // Palet aracın altına girdikten sonra
                                  // aracın yukarı kaldırmasını sağlayan fonksiyon
  {
  output_high(pin_b0);            // Yukarı/Aşağı hareketi sağlayan motor için yön
  for (i=0;i<=356;i++)            // i <= 356 olduğu sürece döngü devam eder
                                  // Döngü bittiğinde araç yukarı kaldırılmıştır
    {
      output_high(pin_b1);
      delay_us(750);
      output_low(pin_b1);
      delay_us(750);
    }
  }
//*****************************************************************************************
void indir()                      // Aracın yere indirilmesini sağlayan fonksiyon
  {
  output_low(pin_b0);             // Yukarı/Aşağı hareketi sağlayan motor için yön
  for (i=0;i<=356;i++)            // i <= 356 olduğu sürece döngü devam eder
                                  // Döngü bittiğinde araç aşağı indirilmiştir
    {
```

EK-1. (devam) Mikrodenetleyici yazılımı

```
        output_high(pin_b1);
        delay_us(750);
        output_low(pin_b1);
        delay_us(750);
      }
  }
//*************************************************************************************
void pulse_1800 ()                  // 1800 Clock palsi sağlayan fonksiyon
  {
  for (i=0;i<=1800;i++)             // i <= 1800 olduğu sürece döngü devam eder
     {
        output_high(pin_b3);
        delay_us(750);
        output_low(pin_b3);
        delay_us(750);
     }
  }
//*************************************************************************************
void pulse_1900 ()                  // 1900 Clock palsi sağlayan fonksiyon.
  {
  for (i=0;i<=1900;i++)             // i <= 1900 olduğu sürece döngü devam eder
     {
        output_high(pin_b3);
        delay_us(750);
        output_low(pin_b3);
        delay_us(750);
     }
  }
//*************************************************************************************
void pulse_3700 ()                  // 3700 Clock palsi sağlayan fonksiyon
  {
  for (i=0;i<=3700;i++)             // i <= 3700 olduğu sürece döngü devam eder
```

EK-1. (devam) Mikrodenetleyici yazılımı

```
    {
      output_high(pin_b3);
      delay_us(750);
      output_low(pin_b3);
      delay_us(750);
    }
  }
//*************************************************************************************************
void pulse_3800 ()                  // 3800 Clock palsi sağlayan fonksiyon
  {
  for (i=0;i<=3800;i++)             // i <= 3800 olduğu sürece döngü devam eder
    {
      output_high(pin_b3);
      delay_us(750);
      output_low(pin_b3);
      delay_us(750);
    }
  }
//*************************************************************************************************
void pulse_5600 ()                  // 5600 Clock palsi sağlayan fonksiyon
  {
  for (i=0;i<=5600;i++)             // i <= 5600 olduğu sürece döngü devam eder
    {
      output_high(pin_b3);
      delay_us(750);
      output_low(pin_b3);
      delay_us(750);
    }
  }
//*************************************************************************************************
void a4den_a3e_koy ()               // Aracı A4'den alıp A3'e yerleştiren fonksiyon
  {
```

EK-1. (devam) Mikrodenetleyici yazılımı

```
  ileri();                    // ileri fonksiyonu ile palet aracın altına girer
  kaldir();                   // kaldır fonk. ile aracın tekerlekleri yerden kesilir
  geri ();                    // geri fonksiyonu ile araç bölmeden çıkarılır
  output_low(pin_b2);         // Sağ/Sol hareketi sağlayan motor için yön
  pulse_1900 ();              // pulse_1900 fonk. ile paletin yön bilgisine göre
                              // sola dönerek A3'ün önüne gelmesi sağlanır
  ileri ();                   // ileri fonk. ile araç A3 bölmesi içine sokulur
  indir ();                   // indir fonksiyonu ile araç yere indirilir
  geri ();                    // geri fonksiyonu ile paletin geri gelmesi sağlanır
  output_high(pin_b2);        // Sağ/Sol hareketi sağlayan motor için yön
  pulse_1900 ();              // pulse_1900 fonk. ile paletin yön bilgisine göre
                              // sağa dönerek A4'ün önüne gelmesi sağlanır
  }
//*****************************************************************************************
void a3den_a4e_koy ()         // Aracı A3'den alıp A4'e yerleştiren fonksiyon
  {
  output_low(pin_b2);         // Sağ/Sol hareketi sağlayan motor için yön
  pulse_1900 ();              // pulse_1900 fonk. ile paletin yön bilgisine göre
                              // sola dönerek A3'ün önüne gelmesi sağlanır
  ileri();                    // ileri fonksiyonu ile palet aracın altına girer
  kaldir();                   // kaldır fonk. ile aracın tekerlekleri yerden kesilir
  geri ();                    // geri fonksiyonu ile araç bölmeden çıkarılır
  output_high(pin_b2);        // Sağ/Sol hareketi sağlayan motor için yön
  pulse_1900 ();              // pulse_1900 fonk. ile paletin yön bilgisine göre
                              // sağa dönerek A4'ün önüne gelmesi sağlanır
  ileri ();                   // ileri fonk. ile araç A4 bölmesi içine sokulur
  indir ();                   // indir fonksiyonu ile araç yere indirilir
  geri ();                    // geri fonksiyonu ile paletin geri gelmesi sağlanır
  }
//*****************************************************************************************
void a4den_a2ye_koy ()        // Aracı A4'den alıp A2'ye yerleştiren fonksiyon
  {
```

EK-1. (devam) Mikrodenetleyici yazılımı

```
  ileri();                    // ileri fonksiyonu ile palet aracın altına girer
  kaldir();                   // kaldır fonk. ile aracın tekerlekleri yerden kesilir
  geri ();                    // geri fonksiyonu ile araç bölmeden çıkarılır.
  output_low(pin_b2);         // Sağ/Sol hareketi sağlayan motor için yön
  pulse_3800 ();              // pulse_3800 fonk. ile paletin yön bilgisine göre
                              // sola dönerek A2'nin önüne gelmesi sağlanır
  ileri ();                   // ileri fonk. ile araç A2 bölmesi içine sokulur
  indir ();                   // indir fonksiyonu ile araç yere indirilir
  geri ();                    // geri fonksiyonu ile paletin geri gelmesi sağlanır
  output_high(pin_b2);        // Sağ/Sol hareketi sağlayan motor için yön
  pulse_3800 ();              // pulse_3800 fonk. ile paletin yön bilgisine göre
                              // sağa dönerek A4'ün önüne gelmesi sağlanır
  }
//*****************************************************************************************
void a2den_a4e_koy ()         // Aracı A2'den alıp A4'e yerleştiren fonksiyon.
  {
  output_low(pin_b2);         // Sağ/Sol hareketi sağlayan motor için yön
  pulse_3800 ();              // pulse_3800 fonk. ile paletin yön bilgisine göre
                              // sola dönerek A2'nin önüne gelmesi sağlanır
  ileri();                    // ileri fonksiyonu ile palet aracın altına girer
  kaldir();                   // kaldır fonk. ile aracın tekerlekleri yerden kesilir
  geri ();                    // geri fonksiyonu ile araç bölmeden çıkarılır.
  output_high(pin_b2);        // Sağ/Sol hareketi sağlayan motor için yön
  pulse_3800 ();              // pulse_3800 fonk. ile paletin yön bilgisine göre
                              // sağa dönerek A4'ün önüne gelmesi sağlanır.
  ileri ();                   // ileri fonk. ile araç A4 bölmesi içine sokulur.
  indir ();                   // indir fonksiyonu ile araç yere indirilir.
  geri ();                    // geri fonksiyonu ile paletin geri gelmesi sağlanır.
  }
//*****************************************************************************************
void a4den_a1e_koy ()         // Aracı A4'den alıp A1'e yerleştiren fonksiyon
  {
```

EK-1. (devam) Mikrodenetleyici yazılımı

```
  ileri();                    // ileri fonksiyonu ile palet aracın altına girer
  kaldir();                   // kaldır fonk. ile aracın tekerlekleri yerden kesilir
  geri ();                    // geri fonksiyonu ile araç bölmeden çıkarılır
  output_low(pin_b2);         // Sağ/Sol hareketi sağlayan motor için yön
  pulse_5600 ();              // pulse_5600 fonk. ile paletin yön bilgisine göre
                              // sola dönerek A1'ün önüne gelmesi sağlanır
  ileri ();                   // ileri fonk. ile araç A1 bölmesi içine sokulur
  indir ();                   // indir fonksiyonu ile araç yere indirilir
  geri ();                    // geri fonksiyonu ile paletin geri gelmesi sağlanır
  output_high(pin_b2);        // Sağ/Sol hareketi sağlayan motor için yön
  pulse_5600 ();              // pulse_5600 fonk. ile paletin yön bilgisine göre
                               // sağa dönerek A4'ün önüne gelmesi sağlanır
  }
//*************************************************************************************
void a1den_a4e_koy ()         // Aracı A1'den alıp A4'e yerleştiren fonksiyon
  {
  output_low(pin_b2);         // Sağ/Sol hareketi sağlayan motor için yön
  pulse_5600 ();              // pulse_5600 fonk. ile paletin yön bilgisine göre
                              // sola dönerek A1'in önüne gelmesi sağlanır
  ileri();                    // ileri fonksiyonu ile palet aracın altına girer
  kaldir();                    // kaldır fonk. ile aracın tekerlekleri yerden kesilir
  geri ();                    // geri fonksiyonu ile araç bölmeden çıkarılır
  output_high(pin_b2);        // Sağ/Sol hareketi sağlayan motor için yön
  pulse_5600 ();              // pulse_5600 fonk.  ile paletin yön bilgisine göre
                              // sağa dönerek A4'ün önüne gelmesi sağlanır
  ileri ();                   // ileri fonk.  ile araç A4 bölmesi içine sokulur
  indir ();                   // indir fonksiyonu ile araç yere indirilir
  geri ();                    // geri fonksiyonu ile paletin geri gelmesi sağlanır
  }
//*************************************************************************************
void a4den_a5e_koy ()         // Aracı A4'den alıp A5'e yerleştiren fonksiyon
  {
```

EK-1. (devam) Mikrodenetleyici yazılımı

```
  ileri();                    // ileri fonksiyonu ile palet aracın altına girer
  kaldir();                   // kaldır fonk.  ile aracın tekerlekleri yerden kesilir
  geri ();                    // geri fonksiyonu ile araç bölmeden çıkarılır
  output_high(pin_b2);        // Sağ/Sol hareketi sağlayan motor için yön
  pulse_1800 ();              // pulse_1800 fonk.  ile paletin yön bilgisine göre
                              // sağa dönerek A5'in önüne gelmesi sağlanır
  ileri ();                   // ileri fonk.  ile araç A5 bölmesi içine sokulur
  indir ();                   // indir fonksiyonu ile araç yere indirilir
  geri ();                    // geri fonksiyonu ile paletin geri gelmesi sağlanır
  output_low(pin_b2);         // Sağ/Sol hareketi sağlayan motor için yön
  pulse_1800 ();              // pulse_1800 fonk.  ile paletin yön bilgisine göre
                               // sola dönerek A4'ün önüne gelmesi sağlanır
  }
//*****************************************************************************************
void a5den_a4e_koy ()         // Aracı A5'den alıp A4'e yerleştiren fonksiyon
  {
  output_high(pin_b2);        // Sağ/Sol hareketi sağlayan motor için yön
  pulse_1800 ();              // pulse_1800 fonk.  ile paletin yön bilgisine göre
                              // sağa dönerek A5'in önüne gelmesi sağlanır
  ileri();                    // ileri fonksiyonu ile palet aracın altına girer
  kaldir();                   // kaldır fonk.  ile aracın tekerlekleri yerden kesilir
  geri ();                    // geri fonksiyonu ile araç bölmeden çıkarılır
  output_low(pin_b2);         // Sağ/Sol hareketi sağlayan motor için yön
  pulse_1800 ();              // pulse_1800 fonk.  ile paletin yön bilgisine göre
                              // sola dönerek A4'ün önüne gelmesi sağlanır
  ileri ();                   // ileri fonk.  ile araç A4 bölmesi içine sokulur
  indir ();                   // indir fonksiyonu ile araç yere indirilir
  geri ();                    // geri fonksiyonu ile paletin geri gelmesi sağlanır
  }
//*****************************************************************************************
void a4den_a6ya_koy ()        // Aracı A4'den alıp A6'ya yerleştiren fonksiyon
  {
```

EK-1. (devam) Mikrodenetleyici yazılımı

```
  ileri();                    // ileri fonksiyonu ile palet aracın altına girer
  kaldir();                   // kaldır fonk.  ile aracın tekerlekleri yerden kesilir
  geri ();                    // geri fonksiyonu ile araç bölmeden çıkarılır

  output_high(pin_b2);        // Sağ/Sol hareketi sağlayan motor için yön
  pulse_3700 ();              // pulse_3700 fonk.  ile paletin yön bilgisine göre
                              // sağa dönerek A6'nın önüne gelmesi sağlanır
  ileri ();                   // ileri fonk. ile araç A6 bölmesi içine sokulur
  indir ();                   // indir fonksiyonu ile araç yere indirilir
  geri ();                    // geri fonksiyonu ile paletin geri gelmesi sağlanır
  output_low(pin_b2);         // Sağ/Sol hareketi sağlayan motor için yön
  pulse_3700 ();              // pulse_3700 fonk. ile paletin yön bilgisine göre
                              // sola dönerek A4'ün önüne gelmesi sağlanır
  }
//*****************************************************************************************
void a6dan_a4e_koy ()         // Aracı A6'dan alıp A4'e yerleştiren fonksiyon
  {
  output_high(pin_b2);        // Sağ/Sol hareketi sağlayan motor için yön
  pulse_3700 ();              // pulse_3700 fonk. ile paletin yön bilgisine göre
                              // sağa dönerek A6'nın önüne gelmesi sağlanır
  ileri();                    // ileri fonksiyonu ile palet aracın altına girer
  kaldir();                   // kaldır fonk. ile aracın tekerlekleri yerden kesilir
  geri ();                    // geri fonksiyonu ile araç bölmeden çıkarılır
  output_low(pin_b2);         // Sağ/Sol hareketi sağlayan motor için yön
  pulse_3700 ();              // pulse_3700 fonk. ile paletin yön bilgisine göre
                              // sola dönerek A4'ün önüne gelmesi sağlanır
  ileri ();                   // ileri fonk. ile araç A4 bölmesi içine sokulur
  indir ();                   // indir fonksiyonu ile araç yere indirilir
  geri ();                    // geri fonksiyonu ile paletin geri gelmesi sağlanır
  }
//*****************************************************************************************
void a4den_a7ye_koy ()        // Aracı A4'den alıp A7'ye yerleştiren fonksiyon
```

EK-1. (devam) Mikrodenetleyici yazılımı

```
  {
  ileri();                    // ileri fonksiyonu ile palet aracın altına girer
  kaldir();                   // kaldır fonk. ile aracın tekerlekleri yerden kesilir
  geri ();                    // geri fonksiyonu ile araç bölmeden çıkarılır
  output_high(pin_b2);        // Sağ/Sol hareketi sağlayan motor için yön
  pulse_5600 ();              // pulse_5600 fonk. ile paletin yön bilgisine göre
                              // sağa dönerek A7'nin önüne gelmesi sağlanır
  ileri ();                   // ileri fonk. ile araç A7 bölmesi içine sokulur
  indir ();                   // indir fonksiyonu ile araç yere indirilir
  geri ();                    // geri fonk. ile paletin geri gelmesi sağlanır
  output_low(pin_b2);         // Sağ/Sol hareketi sağlayan motor için yön
  pulse_5600 ();              // pulse_5600 fonk. ile paletin yön bilgisine göre
                              // sola dönerek A4'ün önüne gelmesi sağlanır
  }
//*****************************************************************************************
void a7den_a4e_koy ()         // Aracı A7'den alıp A4'e yerleştiren fonksiyon
  {
  output_high(pin_b2);        // Sağ/Sol hareketi sağlayan motor için yön
  pulse_5600 ();              // pulse_5600 fonk. ile paletin yön bilgisine göre
                              // sağa dönerek A7'nin önüne gelmesi sağlanır
  ileri();                    // ileri fonksiyonu ile palet aracın altına girer
  kaldir();                   // kaldır fonk. ile aracın tekerlekleri yerden kesilir
  geri ();                    // geri fonksiyonu ile araç bölmeden çıkarılır
  output_low(pin_b2);         // Sağ/Sol hareketi sağlayan motor için yön
  pulse_5600 ();              // pulse_5600 fonk. ile paletin yön bilgisine göre
                              // sola dönerek A4'ün önüne gelmesi sağlanır
  ileri ();                   // ileri fonk. ile araç A4 bölmesi içine sokulur
  indir ();                   // indir fonksiyonu ile araç yere indirilir
  geri ();                    // geri fonksiyonu ile paletin geri gelmesi sağlanır
  }
//*****************************************************************************************
void a4den_b4e_koy ()         // Aracı A4'den alıp B4'e yerleştiren fonksiyon
```

EK-1. (devam) Mikrodenetleyici yazılımı

```
  {
  ileri();                    // ileri fonksiyonu ile palet aracın altına girer
  kaldir();                   // kaldır fonk. ile aracın tekerlekleri yerden kesilir
  geri ();                    // geri fonksiyonu ile araç bölmeden çıkarılır
  kat_cik ();                 // kat_cik fonksiyonu ile palet bir üst kata çıkar

  ileri ();                   // ileri fonk. ile araç B4 bölmesi içine sokulur
  indir ();                   // indir fonksiyonu ile araç yere indirilir
  geri ();                    // geri fonksiyonu ile paletin geri gelmesi sağlanır
  kat_in ();                  // kat_in fonksiyonu ile palet bir alt kata inerek
                              // A4 bölmenin önüne gelir
  }
//*********************************************************************************************
void b4den_a4e_koy ()         // Aracı B4'den alıp A4'e yerleştiren fonksiyon
  {
  kat_cik ();                 // kat_cik fonksiyonu ile palet bir üst kata çıkar
  ileri();                    // ileri fonksiyonu ile palet aracın altına girer
  kaldir();                   // kaldır fonk. ile aracın tekerlekleri yerden kesilir
  geri ();                    // geri fonksiyonu ile araç bölmeden çıkarılır
  kat_in ();                  // kat_in fonksiyonu ile palet bir alt kata inerek
                              // A4 bölmenin önüne gelir
  ileri ();                   // ileri fonk. ile araç A4 bölmesi içine sokulur
  indir ();                   // indir fonksiyonu ile araç yere indirilir
  geri ();                    // geri fonksiyonu ile paletin geri gelmesi sağlanır
  }
//*********************************************************************************************
void a4den_b3e_koy ()         // Aracı A4'den alıp B3'e yerleştiren fonksiyon
  {
  ileri();                    // ileri fonksiyonu ile palet aracın altına girer
  kaldir();                   // kaldır fonk. ile aracın tekerlekleri yerden kesilir
  geri ();                    // geri fonksiyonu ile araç bölmeden çıkarılır
  kat_cik ();                 // kat_cik fonksiyonu ile palet bir üst kata çıkar
```

EK-1. (devam) Mikrodenetleyici yazılımı

```
  output_low(pin_b2);          // Sağ/Sol hareketi sağlayan motor için yön
  pulse_1900 ();               // pulse_1900 fonk. ile paletin yön bilgisine göre
                               // sola dönerek B3'ün önüne gelmesi sağlanır
  ileri ();                    // ileri fonk. ile araç B3 bölmesi içine sokulur
  indir ();                    // indir fonksiyonu ile araç yere indirilir
  geri ();                     // geri fonksiyonu ile paletin geri gelmesi sağlanır
  output_high(pin_b2);         // Sağ/Sol hareketi sağlayan motor için yön
  pulse_1900 ();               // pulse_1900 fonk. ile paletin yön bilgisine göre
                               // sağa dönmesi sağlanır
  kat_in ();                   // kat_in fonksiyonu ile palet bir alt kata inerek
                               // A4 bölmenin önüne gelir
  }
//*****************************************************************************************
void b3den_a4e_koy ()          // Aracı B3'den alıp A4'e yerleştiren fonksiyon
  {
  kat_cik ();                  // kat_cik fonksiyonu ile palet bir üst kata çıkar
  output_low(pin_b2);          // Sağ/Sol hareketi sağlayan motor için yön
  pulse_1900 ();               // pulse_1900 fonk. ile paletin yön bilgisine göre
                               // sola dönerek B3'ün önüne gelmesi sağlanır
  ileri();                     // ileri fonksiyonu ile palet aracın altına girer
  kaldir();                    // kaldır fonk. ile aracın tekerlekleri yerden kesilir
  geri ();                     // geri fonksiyonu ile araç bölmeden çıkarılır
  output_high(pin_b2);         // Sağ/Sol hareketi sağlayan motor için yön
  pulse_1900 ();               // pulse_1900 fonk. ile paletin yön bilgisine göre
                               // sağa dönmesi sağlanır
  kat_in ();                   // kat_in fonksiyonu ile palet bir alt kata inerek
                               // A4 bölmenin önüne gelir
  ileri ();                    // ileri fonk. ile araç A4 bölmesi içine sokulur
  indir ();                    // indir fonksiyonu ile araç yere indirilir
  geri ();                     // geri fonksiyonu ile paletin geri gelmesi sağlanır
  }
```

EK-1. (devam) Mikrodenetleyici yazılımı

```
//*************************************************************************************************
void a4den_b2ye_koy ()          // Aracı A4'den alıp B2'ye yerleştiren fonksiyon
  {
  ileri();                      // ileri fonksiyonu ile palet aracın altına girer
  kaldir();                     // kaldır fonk. ile aracın tekerlekleri yerden kesilir
  geri ();                      // geri fonksiyonu ile araç bölmeden çıkarılır
  kat_cik ();                   // kat_cik fonksiyonu ile palet bir üst kata çıkar
  output_low(pin_b2);           // Sağ/Sol hareketi sağlayan motor için yön
  pulse_3800 ();                // pulse_3800 fonk. ile paletin yön bilgisine göre
                                // sola dönerek B2'nin önüne gelmesi sağlanır
  ileri ();                     // ileri fonk. ile araç B2 bölmesi içine sokulur
  indir ();                     // indir fonksiyonu ile araç yere indirilir
  geri ();                      // geri fonksiyonu ile paletin geri gelmesi sağlanır
  output_high(pin_b2);          // Sağ/Sol hareketi sağlayan motor için yön
  pulse_3800 ();                // pulse_3800 fonk. ile paletin yön bilgisine göre
                                // sağa dönmesi sağlanır
  kat_in ();                    // kat_in fonksiyonu ile palet bir alt kata inerek
                                // A4 bölmenin önüne gelir
  }
//*************************************************************************************************
void b2den_a4e_koy ()           // Aracı B2'den alıp A4'e yerleştiren fonksiyon
  {
  kat_cik ();                   // kat_cik fonksiyonu ile palet bir üst kata çıkar
  output_low(pin_b2);           // Sağ/Sol hareketi sağlayan motor için yön
  pulse_3800 ();                // pulse_3800 fonk. ile paletin yön bilgisine göre
                                // sola dönerek B2'nin önüne gelmesi sağlanır
  ileri();                      // ileri fonksiyonu ile palet aracın altına girer
  kaldir();                     // kaldır fonk. ile aracın tekerlekleri yerden kesilir
  geri ();                      // geri fonksiyonu ile araç bölmeden çıkarılır
  output_high(pin_b2);          // Sağ/Sol hareketi sağlayan motor için yön
  pulse_3800 ();                // pulse_3800 fonk. ile paletin yön bilgisine göre
                                // sağa dönmesi sağlanır
```

EK-1. (devam) Mikrodenetleyici yazılımı

```
   kat_in ();                    // kat_in fonksiyonu ile palet bir alt kata inerek
                                 // A4 bölmenin önüne gelir
   ileri ();                     // ileri fonk. ile araç A4 bölmesi içine sokulur
   indir ();                     // indir fonksiyonu ile araç yere indirilir
   geri ();                      // geri fonksiyonu ile paletin geri gelmesi sağlanır
   }
//*************************************************************************************
void a4den_b1e_koy ()            // Aracı A4'den alıp B1'e yerleştiren fonksiyon
   {
   ileri();                      // ileri fonksiyonu ile palet aracın altına girer
   kaldir();                     // kaldır fonk. ile aracın tekerlekleri yerden kesilir
   geri ();                      // geri fonksiyonu ile araç bölmeden çıkarılır
   kat_cik ();                   // kat_cik fonksiyonu ile palet bir üst kata çıkar
   output_low(pin_b2);           // Sağ/Sol hareketi sağlayan motor için yön
   pulse_5600 ();                // pulse_5600 fonk. ile paletin yön bilgisine göre
                                 // sola dönerek B1'in önüne gelmesi sağlanır
   ileri ();                     // ileri fonk. ile araç B1 bölmesi içine sokulur
   indir ();                     // indir fonksiyonu ile araç yere indirilir
   geri ();                      // geri fonksiyonu ile paletin geri gelmesi sağlanır
   output_high(pin_b2);          // Sağ/Sol hareketi sağlayan motor için yön
   pulse_5600 ();                // pulse_5600 fonk. ile paletin yön bilgisine göre
                                 // sağa dönmesi sağlanır
   kat_in ();                    // kat_in fonksiyonu ile palet bir alt kata inerek
                                 // A4 bölmenin önüne gelir
   }
//*************************************************************************************
void b1den_a4e_koy ()            // Aracı B1'den alıp A4'e yerleştiren fonksiyon
   {
   kat_cik ();                   // kat_cik fonksiyonu ile palet bir üst kata çıkar
   output_low(pin_b2);           // Sağ/Sol hareketi sağlayan motor için yön
   pulse_5600 ();                // pulse_5600 fonk. ile paletin yön bilgisine göre
                                 // sola dönerek B1'in önüne gelmesi sağlanır
```

EK-1. (devam) Mikrodenetleyici yazılımı

```
  ileri();                    // ileri fonksiyonu ile palet aracın altına girer
  kaldir();                   // kaldır fonk. ile aracın tekerlekleri yerden kesilir
  geri ();                    // geri fonksiyonu ile araç bölmeden çıkarılır
  output_high(pin_b2);        // Sağ/Sol hareketi sağlayan motor için yön
  pulse_5600 ();              // pulse_5600 fonk. ile paletin yön bilgisine göre
                              // sağa dönmesi sağlanır
  kat_in ();                  // kat_in fonksiyonu ile palet bir alt kata inerek
                              // A4 bölmenin önüne gelir
  ileri ();                   // ileri fonk. ile araç A4 bölmesi içine sokulur
  indir ();                   // indir fonksiyonu ile araç yere indirilir
  geri ();                    // geri fonksiyonu ile paletin geri gelmesi sağlanır
  }
//*****************************************************************************************
void a4den_b5e_koy ()         // Aracı A4'den alıp B5'e yerleştiren fonksiyon
  {
  ileri();                    // ileri fonksiyonu ile palet aracın altına girer
  kaldir();                   // kaldır fonk. ile aracın tekerlekleri yerden kesilir
  geri ();                    // geri fonksiyonu ile araç bölmeden çıkarılır
  kat_cik ();                 // kat_cik fonksiyonu ile palet bir üst kata çıkar
  output_high(pin_b2);        // Sağ/Sol hareketi sağlayan motor için yön
  pulse_1800 ();              // pulse_1800 fonk. ile paletin yön bilgisine göre
                              // sağa dönerek B5'in önüne gelmesi sağlanır
  ileri ();                   // ileri fonk. ile araç B5 bölmesi içine sokulur
  indir ();                   // indir fonksiyonu ile araç yere indirilir
  geri ();                    // geri fonksiyonu ile paletin geri gelmesi sağlanır
  output_low(pin_b2);         // Sağ/Sol hareketi sağlayan motor için yön
  pulse_1800 ();              // pulse_1800 fonk. ile paletin yön bilgisine göre
                              // sola dönmesi sağlanır
  kat_in ();                  // kat_in fonksiyonu ile palet bir alt kata inerek
                              // A4 bölmenin önüne gelir
  }
```

EK-1. (devam) Mikrodenetleyici yazılımı

```
//*************************************************************************************************
void b5den_a4e_koy ()          // Aracı B5'den alıp A4'e yerleştiren fonksiyon
  {
  kat_cik ();                  // kat_cik fonksiyonu ile palet bir üst kata çıkar
  output_high(pin_b2);         // Sağ/Sol hareketi sağlayan motor için yön
  pulse_1800 ();               // pulse_1800 fonk. ile paletin yön bilgisine göre
                               // sağa dönerek B5'in önüne gelmesi sağlanır
  ileri();                     // ileri fonksiyonu ile palet aracın altına girer
  kaldir();                    // kaldır fonk. ile aracın tekerlekleri yerden kesilir
  geri ();                     // geri fonksiyonu ile araç bölmeden çıkarılır
  output_low(pin_b2);          // Sağ/Sol hareketi sağlayan motor için yön
  pulse_1800 ();               // pulse_1800 fonk. ile paletin yön bilgisine göre
                               // sola dönmesi sağlanır
  kat_in ();                   // kat_in fonksiyonu ile palet bir alt kata inerek
                               // A4 bölmenin önüne gelir
  ileri ();                    // ileri fonk. ile araç A4 bölmesi içine sokulur
  indir ();                    // indir fonksiyonu ile araç yere indirilir
  geri ();                     // geri fonksiyonu ile paletin geri gelmesi sağlanır
  }
//*************************************************************************************************
void a4den_b6ya_koy ()         // Aracı A4'den alıp B6'ya yerleştiren fonksiyon
  {
  ileri();                     // ileri fonksiyonu ile palet aracın altına girer
  kaldir();                    // kaldır fonk. ile aracın tekerlekleri yerden kesilir
  geri ();                     // geri fonksiyonu ile araç bölmeden çıkarılır
  kat_cik ();                  // kat_cik fonksiyonu ile palet bir üst kata çıkar
  output_high(pin_b2);         // Sağ/Sol hareketi sağlayan motor için yön
  pulse_3700 ();               // pulse_3700 fonk. ile paletin yön bilgisine göre
                               // sağa dönerek B6'nın önüne gelmesi sağlanır
  ileri ();                    // ileri fonk. ile araç B6 bölmesi içine sokulur
  indir ();                    // indir fonksiyonu ile araç yere indirilir
  geri ();                     // geri fonksiyonu ile paletin geri gelmesi sağlanır
```

EK-1. (devam) Mikrodenetleyici yazılımı

```
  output_low(pin_b2);          // Sağ/Sol hareketi sağlayan motor için yön
  pulse_3700 ();               // pulse_3700 fonk. ile paletin yön bilgisine göre
                               // sola dönmesi sağlanır
  kat_in ();                   // kat_in fonksiyonu ile palet bir alt kata inerek
                               // A4 bölmenin önüne gelir
  }
//*****************************************************************************************
void b6dan_a4e_koy ()          // Aracı B6'dan alıp A4'e yerleştiren fonksiyon
  {
  kat_cik ();                  // kat_cik fonksiyonu ile palet bir üst kata çıkar
  output_high(pin_b2);         // Sağ/Sol hareketi sağlayan motor için yön
  pulse_3700 ();               // pulse_3700 fonk. ile paletin yön bilgisine göre
                               // sağa dönerek B6'nın önüne gelmesi sağlanır
  ileri();                     // ileri fonksiyonu ile palet aracın altına girer
  kaldir();                    // kaldır fonk. ile aracın tekerlekleri yerden kesilir
  geri ();                     // geri fonksiyonu ile araç bölmeden çıkarılır
  output_low(pin_b2);          // Sağ/Sol hareketi sağlayan motor için yön
  pulse_3700 ();               // pulse_3700 fonk. ile paletin yön bilgisine göre
                               // sola dönmesi sağlanır
  kat_in ();                   // kat_in fonksiyonu ile palet bir alt kata inerek
                               // A4 bölmenin önüne gelir
  ileri ();                    // ileri fonk. ile araç A4 bölmesi içine sokulur
  indir ();                    // indir fonksiyonu ile araç yere indirilir
  geri ();                     // geri fonksiyonu ile paletin geri gelmesi sağlanır
  }
//*****************************************************************************************
void a4den_b7ye_koy ()         // Aracı A4'den alıp B7'ye yerleştiren fonksiyon
  {
  ileri();                     // ileri fonksiyonu ile palet aracın altına girer
  kaldir();                    // kaldır fonk. ile aracın tekerlekleri yerden kesilir
  geri ();                     // geri fonksiyonu ile araç bölmeden çıkarılır
  kat_cik ();                  // kat_cik fonksiyonu ile palet bir üst kata çıkar
```

EK-1. (devam) Mikrodenetleyici yazılımı

```
  output_high(pin_b2);          // Sağ/Sol hareketi sağlayan motor için yön
  pulse_5600 ();                // pulse_5600 fonk. ile paletin yön bilgisine göre
                                // sağa dönerek B7'nin önüne gelmesi sağlanır
  ileri ();                     // ileri fonk. ile araç B7 bölmesi içine sokulur
  indir ();                     // indir fonksiyonu ile araç yere indirilir
  geri ();                      // geri fonksiyonu ile paletin geri gelmesi sağlanır
  output_low(pin_b2);           // Sağ/Sol hareketi sağlayan motor için yön
  pulse_5600 ();                // pulse_5600 fonk. ile paletin yön bilgisine göre
                                // sola dönmesi sağlanır
  kat_in ();                    // kat_in fonksiyonu ile palet bir alt kata inerek
                                // A4 bölmenin önüne gelir
  }
//*****************************************************************************************
void b7den_a4e_koy ()           // Aracı B7'den alıp A4'e yerleştiren fonksiyon
  {
  kat_cik ();                   // kat_cik fonksiyonu ile palet bir üst kata çıkar
  output_high(pin_b2);          // Sağ/Sol hareketi sağlayan motor için yön
  pulse_5600 ();                // pulse_5600 fonk. ile paletin yön bilgisine göre
                                // sağa dönerek B7'nin önüne gelmesi sağlanır
  ileri();                      // ileri fonksiyonu ile palet aracın altına girer
  kaldir();                     // kaldır fonk. ile aracın tekerlekleri yerden kesilir
  geri ();                      // geri fonksiyonu ile araç bölmeden çıkarılır
  output_low(pin_b2);           // Sağ/Sol hareketi sağlayan motor için yön bilgisi
  pulse_5600 ();                // pulse_5600 fonk. ile paletin yön bilgisine göre
                                // sola dönmesi sağlanır
  kat_in ();                    // kat_in fonksiyonu ile palet bir alt kata inerek
                                // A4 bölmenin önüne gelir
  ileri ();                     // ileri fonk. ile araç A4 bölmesi içine sokulur
  indir ();                     // indir fonksiyonu ile araç yere indirilir
  geri ();                      // geri fonksiyonu ile paletin geri gelmesi sağlanır
  }
```

EK-1. (devam) Mikrodenetleyici yazılımı

```
//-----------------------------------RS232 KESMESİ-------------------------------------------
#int_rda                          // Seri iletişim kesmesi
void rs232_kesmesi()              // Kesme oluştuğunda yapılacak işlemler
  {
  disable_interrupts(int_rda);    // int_rda kesmesi pasif edilir
  veri=getc();                    // gelen bilgi alınır ve "veri" değişkenine aktarılır
  if (veri=='W')                  // eğer gelen bilgi "W" ise araç girişi yapılacaktır
    {
    if(input(PIN_D3)==1 )         // Eğer A3 bölmesi boş ise;
      {
        a4den_a3e_koy ();         // Aracı A3'e park et
        veri=' ';                 // "veri" değişkenini temizle
        putc ('C');               // Park bölmesine ait bilgiyi gönder
      }
    else if(input(PIN_D4)==1 )    // Eğer A5 bölmesi boş ise;
      {
        a4den_a5e_koy ();         // Aracı A5'e park et
        veri=' ';                 // "veri" değişkenini temizle
        putc ('D');               // Park bölmesine ait bilgiyi gönder
      }
    else if(input(PIN_C3)==1 )    // Eğer B4 bölmesi boş ise;
      {
        a4den_b4e_koy ();         // Aracı B4'e park et
        veri=' ';                 // "veri" değişkenini temizle
        putc ('J');               // Park bölmesine ait bilgiyi gönder
      }
    else if(input(PIN_D2)==1 )    // Eğer A2 bölmesi boş ise;
      {
        a4den_a2ye_koy ();        // Aracı A2'ye park et
        veri=' ';                 // "veri" değişkenini temizle
        putc ('B');               // Park bölmesine ait bilgiyi gönder
      }
```

EK-1. (devam) Mikrodenetleyici yazılımı

```
else if(input(PIN_D5)==1 )     // Eğer A6 bölmesi boş ise;
  {
    a4den_a6ya_koy ();         // Aracı A6'ya park et
    veri=' ';                  // "veri" değişkenini temizle
    putc ('E');                // Park bölmesine ait bilgiyi gönder
  }
else if(input(PIN_C2)==1 )     // Eğer B3 bölmesi boş ise;
  {
    a4den_b3e_koy ();          // Aracı B3'e park et
    veri=' ';                  // "veri" değişkenini temizle
    putc ('I');                // Park bölmesine ait bilgiyi gönder
  }
else if(input(PIN_C4)==1 )     // Eğer B5 bölmesi boş ise;
  {
    a4den_b5e_koy ();          // Aracı B5'e park et
    veri=' ';                  // "veri" değişkenini temizle
    putc ('K');                // Park bölmesine ait bilgiyi gönder
  }
else if(input(PIN_D1)==1 )     // Eğer A1 bölmesi boş ise;
  {
    a4den_a1e_koy ();          // Aracı A1'e park et
    veri=' ';                  // "veri" değişkenini temizle
    putc ('A');                // Park bölmesine ait bilgiyi gönder
  }
else if(input(PIN_D6)==1 )     // Eğer A7 bölmesi boş ise;
  {
    a4den_a7ye_koy ();         // Aracı A7'ye park et
    veri=' ';                  // "veri" değişkenini temizle
    putc ('F');                // Park bölmesine ait bilgiyi gönder
  }
else if(input(PIN_C1)==1 )     // Eğer B2 bölmesi boş ise;
  {
```

EK-1. (devam) Mikrodenetleyici yazılımı

```
        a4den_b2ye_koy ();        // Aracı B2'ye park et
        veri=' ';                 // "veri" değişkenini temizle
        putc ('H');               // Park bölmesine ait bilgiyi gönder
      }
    else if(input(PIN_C5)==1 )    // Eğer B6 bölmesi boş ise;
      {
        a4den_b6ya_koy ();        // Aracı B6'ya park et
        veri=' ';                 // "veri" değişkenini temizle
        putc ('L');               // Park bölmesine ait bilgiyi gönder
      }
    else if(input(PIN_C0)==1 )    // Eğer B1 bölmesi boş ise;
      {
        a4den_b1e_koy ();         // Aracı B1'e park et
        veri=' ';                 // "veri" değişkenini temizle
        putc ('G');               // Park bölmesine ait bilgiyi gönder
      }
    else if(input(PIN_D7)==1 )    // Eğer B7 bölmesi boş ise;
      {
        a4den_b7ye_koy ();        // Aracı B7'ye park et
        veri=' ';                 // "veri" değişkenini temizle
        putc ('M');               // Park bölmesine ait bilgiyi gönder
      }
    else
      {
        putc ('N');               // Boş yer olmadığını bildirmek için N
                                  // bilgisini gönderir
      }
   }
  if (veri=='A')                  // Eğer gelen bilgi "A" ise;
    {
      a1den_a4e_koy();            // A1 bölmesindeki aracı teslim et
      putc ('W');                 // "W" göndererek işlemin tamamlandığını bildir
```

EK-1. (devam) Mikrodenetleyici yazılımı

```
      t=1;                    // "t" değişkenini 1 yap
    }
  if (veri=='B')              // Eğer gelen bilgi "B" ise;
    {
      a2den_a4e_koy();        // A2 bölmesindeki aracı teslim et
      putc ('W');             // "W" göndererek işlemin tamamlandığını bildir
      t=1;                    // "t" değişkenini 1 yap
    }
  if (veri=='C')              // Eğer gelen bilgi "C" ise;
    {
      a3den_a4e_koy();        // A3 bölmesindeki aracı teslim et
      putc ('W');             // "W" göndererek işlemin tamamlandığını bildir
      t=1;                    // "t" değişkenini 1 yap
    }
  if (veri=='D')              // Eğer gelen bilgi "D" ise;
    {
      a5den_a4e_koy();        // A5 bölmesindeki aracı teslim et
      putc ('W');             // "W" göndererek işlemin tamamlandığını bildir
      t=1;                    // "t" değişkenini 1 yap
    }
  if (veri=='E')              // Eğer gelen bilgi "E" ise;
    {
      a6dan_a4e_koy();        // A6 bölmesindeki aracı teslim et
      putc ('W');             // "W" göndererek işlemin tamamlandığını bildir
      t=1;                    // "t" değişkenini 1 yap
    }
  if (veri=='F')              // Eğer gelen bilgi "F" ise;
    {
      a7den_a4e_koy();        // A7 bölmesindeki aracı teslim et
      putc ('W');             // "W" göndererek işlemin tamamlandığını bildir
      t=1;                    // "t" değişkenini 1 yap
    }
```

EK-1. (devam) Mikrodenetleyici yazılımı

```
  if (veri=='G')                  // Eğer gelen bilgi "G" ise;
    {
      b1den_a4e_koy();            // B1 bölmesindeki aracı teslim et
      putc ('W');                 // "W" göndererek işlemin tamamlandığını bildir
      t=1;                        // "t" değişkenini 1 yap
    }
  if (veri=='H')                  // Eğer gelen bilgi "H" ise;
    {
      b2den_a4e_koy();            // B2 bölmesindeki aracı teslim et
      putc ('W');                 // "W" göndererek işlemin tamamlandığını bildir
      t=1;                        // "t" değişkenini 1 yap
    }
  if (veri=='I')                  // Eğer gelen bilgi "I" ise;
    {
      b3den_a4e_koy();            // B3 bölmesindeki aracı teslim et
      putc ('W');                 // "W" göndererek işlemin tamamlandığını bildir
      t=1;                        // "t" değişkenini 1 yap
    }
  if (veri=='J')                  // Eğer gelen bilgi "J" ise;
    {
      b4den_a4e_koy();            // B4 bölmesindeki aracı teslim et
      putc ('W');                 // "W" göndererek işlemin tamamlandığını bildir
      t=1;                        // "t" değişkenini 1 yap
    }
  if (veri=='K')                  // Eğer gelen bilgi "K" ise;
    {
      b5den_a4e_koy();            // B5 bölmesindeki aracı teslim et
      putc ('W');                 // "W" göndererek işlemin tamamlandığını bildir
      t=1;                        // "t" değişkenini 1 yap
    }
  if (veri=='L')                  // Eğer gelen bilgi "L" ise;
    {
```

EK-1. (devam) Mikrodenetleyici yazılımı

```
      b6dan_a4e_koy();          // B6 bölmesindeki aracı teslim et
      putc ('W');               // "W" göndererek işlemin tamamlandığını bildir
      t=1;                      // "t" değişkenini 1 yap
    }
  if (veri=='M')                // Eğer gelen bilgi "M" ise;
    {
      b7den_a4e_koy();          // B7 bölmesindeki aracı teslim et
      putc ('W');               // "W" göndererek işlemin tamamlandığını bildir
      t=1;                      // "t" değişkenini 1 yap
    }
  }
//*****************************************************************************************
void main()                     // ANA PROGRAM FONKSİYONU
  {
  setup_psp(PSP_DISABLED);          // PSP birimi devre dışı.
  setup_spi(SPI_SS_DISABLED);       // SPI birimi devre dışı
  setup_timer_1(T1_DISABLED);       // T1 zamanlayıcısı devre dışı
  setup_timer_2(T2_DISABLED,0,1);   // T2 zamanlayıcısı devre dışı
  setup_adc_ports(NO_ANALOGS);      // Analog giriş yok
  setup_adc(ADC_OFF);               // ADC birimi devre dışı
  setup_CCP1(CCP_OFF);              // CCP1 birimi devre dışı
  setup_CCP2(CCP_OFF);              // CCP2 birimi devre dışı

  output_b(0x00);               // PortB'yi temizle (adım motorlar)
  set_tris_c(0b11111111);
  home_position();              // Başlangıç konumuna geç
  delay_ms(500);                // 0,5 Sn bekle

  putc('X');                    // Bilg.Programına başla komutunu gönder
  delay_ms(500);                // 0,5 Sn bekle
  veri=' ';                     // "veri" değişkenini temizle
  t=0;                           // "t" değişkenini 0 yap
```

EK-1. (devam) Mikrodenetleyici yazılımı

```
  enable_interrupts(GLOBAL);    // Tüm kesmelere izin ver
  enable_interrupts(int_rda);   // RS232 kesmesine izin ver

kontrol:
  do
  {
    delay_ms(100);
  }
  while (input(PIN_D0)==1);     // A4 bölmesine araç gelene kadar bekle
  delay_us(500);                // 0,5 mSn bekle

  if (t==1)                     // "t" değişkeni 1 ise;
    {
      goto bekle;               // Araç teslimi yapılmıştır, "bekle" etiketine git
    }
  putc('Q');                    // Araç girişi vardır, foto çek komutu gönder
  delay_ms(200);                // 0,2 Sn bekle

bekle:
 do
    {

    }
  while (input(PIN_D0)==0);     // A4 bölmesi dolu olduğu surece bekle
  t=0;                          // "t" değişkenini 0 yap
  enable_interrupts(GLOBAL);    // Tüm kesmelere izin ver
  enable_interrupts(int_rda);   // RS232 kesmesine izin ver
  delay_ms(3000);               // 3 Sn bekle
  goto kontrol;                 // "kontrol" etiketine git
  }
```

EK-2. Bilgisayar yazılımı

```
using System;
using System.Collections;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Data.OleDb;
using System.Drawing;
using System.Text;
using System.Windows.Forms;
using System.Xml;
using System.IO.Ports;
using System.Threading;

namespace Webcam_Test
{

        public partial class Form1 : System.Windows.Forms.Form
    {
        private string gelen;
        private Bitmap bmp;
        MODI.Document insDocument;
                    private WebCam_Capture.WebCamCapture UserControl1;
                    private WebCam_Capture.WebCamCapture WebCamCapture;
                    private System.Windows.Forms.PictureBox pictureBox1;
                    private System.Windows.Forms.Button cmdStart;
                    private System.Windows.Forms.Button cmdStop;
                    private System.Windows.Forms.Button cmdContinue;
                    private System.Windows.Forms.NumericUpDown numCaptureTime;
                    private System.Windows.Forms.Label label1;
        private Button button1;
        private RichTextBox richTextBox1;
        private PictureBox pictureBox2;
        private OpenFileDialog openFileDialog1;
        private SerialPort serialPort1;
        private ComboBox comboBox1;
        private TextBox textBox1;
        private Button button2;
        public ListView listView1;
        private ColumnHeader columnHeader1;
        private ColumnHeader columnHeader2;
        private ColumnHeader columnHeader3;
        private ColumnHeader columnHeader4;
        private ColumnHeader columnHeader5;
        private ColumnHeader columnHeader6;
        private ColumnHeader columnHeader7;
        private Button button3;
        private TextBox textBox2;
```

EK-2. (devam) Bilgisayar yazılımı

```
        private Button button4;
        private Button button5;
        private PictureBox pictureBox4;
        private PictureBox pictureBox3;
        private Label label3;
        private Label label5;
        private Label label4;
        private Label label2;
        private IContainer components;

//*************************************************************************************************
                public Form1()
                {
                        InitializeComponent();
          insDocument = new MODI.Document();
          Control.CheckForIllegalCrossThreadCalls = false;
                }
//*************************************************************************************************

                protected override void Dispose( bool disposing )
                {
                        if( disposing )
                        {
                                if (components != null)
                                {
                                        components.Dispose();
                                }
                        }
                        base.Dispose( disposing );
                }
//*************************************************************************************************

          [STAThread]
                static void Main()
                {
                        Application.Run(new Form1());
                }
//*************************************************************************************************
      OleDbConnection baglanma = new
OleDbConnection("Provider=Microsoft.ACE.OLEDB.12.0;Data Source=
db.accdb");

//*************************************************************************************************
      private void serialPort1_DataReceived(object sender,
System.IO.Ports.SerialDataReceivedEventArgs e)
      {
          if (serialPort1.IsOpen)
          {
```

EK-2. (devam) Bilgisayar yazılımı

```
            serialPort1.Close();
        }
        serialPort1.PortName = comboBox1.SelectedItem.ToString();
        serialPort1.Open();
        gelen = serialPort1.ReadExisting();
        Thread paralel1 = new Thread(new ThreadStart(okuma));
        paralel1.Start();
        paralel1.Join();
    }
//*********************************************************************************************

                private void Form1_Load(object sender, System.EventArgs e)
    {
        if (serialPort1.IsOpen)
        {
            serialPort1.Close();
        }
        foreach (string ComPorts in SerialPort.GetPortNames())
        comboBox1.Items.Add(ComPorts);
        comboBox1.SelectedIndex = 0;
        serialPort1.Open();
        serialPort1.DataReceived                    +=                    new
SerialDataReceivedEventHandler(okuma);
        do
        {

        } while (gelen!=”X”);
        OleDbCommand komut = new OleDbCommand("Select * From kayıtlar ",
baglanma);
        OleDbDataReader okuyucu = null;
        baglanma.Open();
        okuyucu = komut.ExecuteReader();
        listView1.Items.Clear();
        while (okuyucu.Read())
        {
            ListViewItem Liste = new ListViewItem(okuyucu["id"].ToString());
            Liste.SubItems.Add(okuyucu["plaka"].ToString());
            Liste.SubItems.Add(okuyucu["giriş"].ToString());
            Liste.SubItems.Add(okuyucu["port"].ToString());
            Liste.SubItems.Add(okuyucu["çıkış"].ToString());
            Liste.SubItems.Add(okuyucu["ücret"].ToString());
            Liste.SubItems.Add(okuyucu["kullanım"].ToString());
            listView1.Items.Add(Liste);
        }
            baglanma.Close();
                    this.WebCamCapture.CaptureHeight                    =
this.pictureBox1.Height;
                    this.WebCamCapture.CaptureWidth = this.pictureBox1.Width;
```

EK-2. (devam) Bilgisayar yazılımı

```
        this.WebCamCapture.TimeToCapture_milliseconds                    =
(int)this.numCaptureTime.Value;
        this.WebCamCapture.Start(0);
    }
//*************************************************************************************

            private          void          Form1_Closing(object          sender,
System.ComponentModel.CancelEventArgs e)
            {
                this.WebCamCapture.Stop();
            }
//*************************************************************************************
            private    void    WebCamCapture_ImageCaptured(object    source,
WebCam_Capture.WebcamEventArgs e)
            {
                this.pictureBox1.Image = e.WebCamImage;
            }
//*************************************************************************************

            private void cmdStart_Click(object sender, System.EventArgs e)
            {
                this.WebCamCapture.TimeToCapture_milliseconds    =    (int)
this.numCaptureTime.Value;
        this.WebCamCapture.Start(0);
            }
//*************************************************************************************

            private void cmdStop_Click(object sender, System.EventArgs e)
            {
                this.WebCamCapture.Stop();
            }
//*************************************************************************************

            private void cmdContinue_Click(object sender, System.EventArgs e)

            {
                this.WebCamCapture.TimeToCapture_milliseconds    =    (int)
this.numCaptureTime.Value;
        this.WebCamCapture.Start(this.WebCamCapture.FrameNumber);
            }
//*************************************************************************************
    private void button1_Click(object sender, EventArgs e)
    {
        imageprocess process = new imageprocess(pictureBox1.Image);
        pictureBox2.Image = process.kes(110, 90, 120, 60);
        pictureBox2.Image.Save("b.jpg");
        insDocument.Create("b.jpg");
        try
```

EK-2. (devam) Bilgisayar yazılımı

```
        {
            insDocument.OCR(MODI.MiLANGUAGES.miLANG_ENGLISH,      true,
true);
            foreach (MODI.Image insImage in insDocument.Images)
            {
                MODI.Layout insLayout = insImage.Layout;
                string s = insLayout.Text;
                string plate = "";
                for (int i = 0; i < s.Length; i++)
                {
                    if ((Convert.ToInt32(s[i]) >= 48 && Convert.ToInt32(s[i]) <= 57) ||
(Convert.ToInt32(s[i])    >=    65    &&    Convert.ToInt32(s[i])    <=    90)    &&
Convert.ToInt32(s[i]) != 81 && Convert.ToInt32(s[i]) != 87 && Convert.ToInt32(s[i])
!= 88)
                    {
                        plate += s[i];
                    }
                }
                richTextBox1.Text = plate;
            }
            insDocument.Close(false);
        }
        catch (Exception)
        {
            MessageBox.Show("can\'t read");
        }
    }

//*****************************************************************************************
private void okuma(object s, SerialDataReceivedEventArgs e)
    {
            gelen = serialPort1.ReadExisting();
            textBox1.Text = gelen;
        if (gelen=="Q")
        {
            try
            {
            imageprocess process = new imageprocess(pictureBox1.Image);
            pictureBox2.Image = process.kes(110, 90, 120, 60);
            pictureBox2.Image.Save("b.jpg");
            insDocument.Create("b.jpg");
            }
            catch (Exception)
            {
                MessageBox.Show("Yeniden foto çekiliyor");
            }
            try
            {
```

EK-2. (devam) Bilgisayar yazılımı

```
                insDocument.OCR(MODI.MiLANGUAGES.miLANG_ENGLISH,  true,
true);
                foreach (MODI.Image insImage in insDocument.Images)
                {
                   MODI.Layout insLayout = insImage.Layout;
                   string a = insLayout.Text;
                   string plate = "";
                   for (int i = 0; i < a.Length; i++)
                   {
                      if ((Convert.ToInt32(a[i]) >= 48 && Convert.ToInt32(a[i]) <= 57) ||
(Convert.ToInt32(a[i])  >=  65  &&  Convert.ToInt32(a[i])  <=  90)  &&
Convert.ToInt32(a[i]) != 81 && Convert.ToInt32(a[i]) != 87 && Convert.ToInt32(a[i])
!= 88)
                      {
                         plate += a[i];
                      }
                   }
                   richTextBox1.Text = plate;
                }
                insDocument.Close(false);
             }
             catch (Exception)
             {
                MessageBox.Show("Plaka Tanımlanamadı");
             }
          }
       }
//*********************************************************************************************
       private void okuma()
       {
          gelen = serialPort1.ReadExisting();
          textBox1.Text = gelen;
       }
//*********************************************************************************************
       private void textBox1_TextChanged(object sender, EventArgs e)
       {

       }
//*********************************************************************************************
       private void button2_Click(object sender, EventArgs e)
       {
          if (gelen != "Q")
          {
             MessageBox.Show("Giriş İçin Gelen Araç Yok yada Sensör Arızalı");
          }
          else
          {
             serialPort1.Write("W");
```

EK-2. (devam) Bilgisayar yazılımı

```
            do
            {

            } while (gelen != "A" && gelen != "B" && gelen != "C" && gelen != "D" &&
gelen != "E" && gelen != "F" && gelen != "G" && gelen != "H" && gelen != "I" &&
gelen != "J" && gelen != "K" && gelen != "L" && gelen != "M" && gelen != "N");

            if (gelen == "N")
            {
               MessageBox.Show("Boş Park Yerimiz Yoktur");
               richTextBox1.Text = "";
            }
            else
            {

               OleDbCommand kullan = new OleDbCommand("Select count(plaka)
from kayıtlar where plaka = '" + richTextBox1.Text + "'", baglanma);
               baglanma.Open();
               int k_say = (int)kullan.ExecuteScalar();
               baglanma.Close();
               bool cevir = true;
               do
               {
                  if (gelen == "A")
                  {
                     textBox1.Text = "A1";
                     cevir = false;
                     button6.BackColor = System.Drawing.Color.Red;
                  }
                  else if (gelen == "B")
                  {
                     textBox1.Text = "A2";
                     cevir = false;
                     button7.BackColor = System.Drawing.Color.Red;
                  }
                  else if (gelen == "C")
                  {
                     textBox1.Text = "A3";
                     cevir = false;
                     button8.BackColor = System.Drawing.Color.Red;
                  }
                  else if (gelen == "D")
                  {
                     textBox1.Text = "A5";
                     cevir = false;
                     button10.BackColor = System.Drawing.Color.Red;
                  }
                  else if (gelen == "E")
```

EK-2. (devam) Bilgisayar yazılımı

```
{
    textBox1.Text = "A6";
    cevir = false;
    button11.BackColor = System.Drawing.Color.Red;
}
else if (gelen == "F")
{
    textBox1.Text = "A7";
    cevir = false;
    button12.BackColor = System.Drawing.Color.Red;
}
else if (gelen == "G")
{
    textBox1.Text = "B1";
    cevir = false;
    button13.BackColor = System.Drawing.Color.Red;
}
else if (gelen == "H")
{
    textBox1.Text = "B2";
    cevir = false;
    button14.BackColor = System.Drawing.Color.Red;
}
else if (gelen == "I")
{
    textBox1.Text = "B3";
    cevir = false;
    button15.BackColor = System.Drawing.Color.Red;
}
else if (gelen == "J")
{
    textBox1.Text = "B4";
    cevir = false;
    button16.BackColor = System.Drawing.Color.Red;
}
else if (gelen == "K")
{
    textBox1.Text = "B5";
    cevir = false;
    button17.BackColor = System.Drawing.Color.Red;
}
else if (gelen == "L")
{
    textBox1.Text = "B6";
    cevir = false;
    button18.BackColor = System.Drawing.Color.Red;
}
else if (gelen == "M")
```

EK-2. (devam) Bilgisayar yazılımı

```
                {
                    textBox1.Text = "B7";
                    cevir = false;
                    button19.BackColor = System.Drawing.Color.Red;
                }
            } while (cevir);
            DateTime şimdi = Convert.ToDateTime(DateTime.Now.ToShortTimeString());
            string şimdik = şimdi.ToString("MM.dd.yyyy HH:mm:ss");
            şimdik = şimdik.Replace('.', '/');
            OleDbCommand ekle_komutu = new OleDbCommand("Insert Into kayıtlar(plaka, giriş, port, kullanım) VALUES ('" + richTextBox1.Text + "' , #" + şimdik + "#, '" + textBox1.Text + "', " + (k_say + 1) + " ) ", baglanma);
            baglanma.Open();
            ekle_komutu.ExecuteNonQuery();
            baglanma.Close();
            OleDbCommand komut = new OleDbCommand("Select * From kayıtlar ", baglanma);
            OleDbDataReader okuyucu = null;
            baglanma.Open();
            okuyucu = komut.ExecuteReader();
            listView1.Items.Clear();
            while (okuyucu.Read())
            {
                ListViewItem Liste = new ListViewItem(okuyucu["id"].ToString());
                Liste.SubItems.Add(okuyucu["plaka"].ToString());
                Liste.SubItems.Add(okuyucu["giriş"].ToString());
                Liste.SubItems.Add(okuyucu["port"].ToString());
                Liste.SubItems.Add(okuyucu["çıkış"].ToString());
                Liste.SubItems.Add(okuyucu["ücret"].ToString());
                Liste.SubItems.Add(okuyucu["kullanım"].ToString());
                listView1.Items.Add(Liste);
            }
            baglanma.Close();
            MessageBox.Show("Araç " + textBox1.Text + "Slotuna Park Edilmiştir");
            richTextBox1.Text = "";
        }
    }
}


//*****************************************************************************************
private string str;

private void button3_Click(object sender, EventArgs e)
    {
        if (button3.Text=="Çıkış")
```

EK-2. (devam) Bilgisayar yazılımı

```
        {
            label5.Visible=true;
              textBox2.Visible = true;
              button3.Text = "Gönder";
        }
        else if (button3.Text=="Gönder")
        {
            try
             {
             OleDbCommand saat_al = new OleDbCommand("select giriş from kayıtlar where plaka='" + textBox2.Text + "' and çıkış is null ", baglanma);
             baglanma.Open();
             OleDbDataAdapter oku = new OleDbDataAdapter(saat_al);
             DataTable tablo = new DataTable();
             oku.Fill(tablo);
             ArrayList arrayList = new ArrayList();
             foreach (DataRow dtr in tablo.Rows)
             {
                foreach (DataColumn dtc in tablo.Columns)
                {
                   arrayList.Add(dtr[dtc]);
                }
             }
             DateTime saat = Convert.ToDateTime(arrayList[0]);
             DateTime şimdi = Convert.ToDateTime(DateTime.Now.ToShortTimeString());
             string tar = şimdi.ToString("MM.dd.yyyy HH:mm:ss");
             tar = tar.Replace('.', '/');
             TimeSpan fark = (şimdi - saat);
             int ücret = Convert.ToInt16((Math.Ceiling(Convert.ToDouble(fark.Minutes) / 60) + fark.Hours + fark.Days * 24).ToString()) * 5;
             baglanma.Close();
             arrayList.Clear();

               //port seç

             OleDbCommand port_al = new OleDbCommand("select port from kayıtlar where plaka='" + textBox2.Text + "' and çıkış is null ", baglanma);
             baglanma.Open();
             OleDbDataAdapter oku2 = new OleDbDataAdapter(port_al);
             DataTable tablo_port = new DataTable();
             oku2.Fill(tablo_port);
             ArrayList arrayList_port = new ArrayList();
             foreach (DataRow dtr_port in tablo_port.Rows)
             {
                foreach (DataColumn dtc_port in tablo_port.Columns)
                {
```

EK-2. (devam) Bilgisayar yazılımı

```
                arrayList_port.Add(dtr_port[dtc_port]);
            }
        }
        string port_no = Convert.ToString( arrayList_port[0]);

        baglanma.Close();

          // kayıt güncelle
        OleDbCommand cıkar = new OleDbCommand("update kayıtlar set çıkış
=#" + tar + "# , ücret= " + ücret + "  where plaka='" + textBox2.Text + "' and çıkış is
null ", baglanma);
        baglanma.Open();
        cıkar.ExecuteNonQuery();
        baglanma.Close();

            if (port_no == "A1")
            {
                str = "A";
                button6.BackColor = System.Drawing.Color.Lime;
            }
            else if (port_no == "A2")
            {
                str = "B";
                button7.BackColor = System.Drawing.Color.Lime;
            }
            else if (port_no == "A3")
            {
                str = "C";
                button8.BackColor = System.Drawing.Color.Lime;
            }
            else if (port_no == "A5")
            {
                str = "D";
                button10.BackColor = System.Drawing.Color.Lime;
            }
            else if (port_no == "A6")
            {
                str = "E";
                button11.BackColor = System.Drawing.Color.Lime;
            }
            else if (port_no == "A7")
            {
                str = "F";
                button12.BackColor = System.Drawing.Color.Lime;
            }
            else if (port_no == "B1")
            {
                str = "G";
```

EK-2. (devam) Bilgisayar yazılımı

```
                button13.BackColor = System.Drawing.Color.Lime;
            }
            else if (port_no == "B2")
            {
                str = "H";
                button14.BackColor = System.Drawing.Color.Lime;
            }
            else if (port_no == "B3")
            {
                str = "I";
                button15.BackColor = System.Drawing.Color.Lime;
            }
            else if (port_no == "B4")
            {
                str = "J";
                button16.BackColor = System.Drawing.Color.Lime;
            }
            else if (port_no == "B5")
            {
                str = "K";
                button17.BackColor = System.Drawing.Color.Lime;
            }
            else if (port_no == "B6")
            {
                str = "L";
                button18.BackColor = System.Drawing.Color.Lime;
            }
            else if (port_no == "B7")
            {
                str = "M";
                button19.BackColor = System.Drawing.Color.Lime;
            }
            serialPort1.Write(str);
            gelen = "";
            do
            {

            } while (gelen != "W");
         MessageBox.Show(    "Araç Teslim Edilmiştir. Ücret "+ ücret + " TL. "
);
        OleDbCommand komut = new OleDbCommand("Select * From kayıtlar
", baglanma);
        OleDbDataReader okuyucu = null;
        baglanma.Open();
        okuyucu = komut.ExecuteReader();
        listView1.Items.Clear();
        while (okuyucu.Read())
        {
```

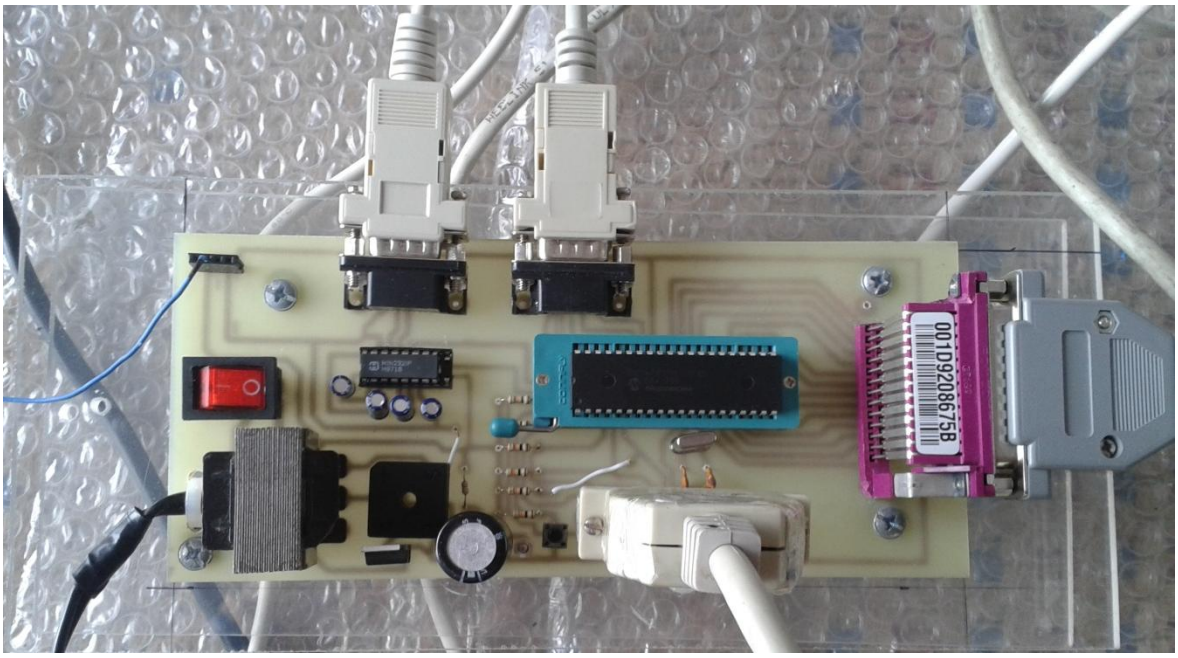
EK-2. (devam) Bilgisayar yazılımı

```
                ListViewItem Liste = new ListViewItem(okuyucu["id"].ToString());
                Liste.SubItems.Add(okuyucu["plaka"].ToString());
                Liste.SubItems.Add(okuyucu["giriş"].ToString());
                Liste.SubItems.Add(okuyucu["port"].ToString());
                Liste.SubItems.Add(okuyucu["çıkış"].ToString());
                Liste.SubItems.Add(okuyucu["ücret"].ToString());
                Liste.SubItems.Add(okuyucu["kullanım"].ToString());
                listView1.Items.Add(Liste);
            }
            baglanma.Close();
            label5.Visible = false;
            textBox2.Visible = false;
            button3.Text = "Çıkış";
            }
            catch (Exception)
            {
                baglanma.Close();
                MessageBox.Show("Hatalı plaka girildi yada plaka kayıtlarda yok");
                textBox2.Visible = false;
                button3.Text = "Çıkış";
            }
         }
        textBox2.Text = "";
    }

//*********************************************************************************************
    private void button5_Click(object sender, EventArgs e)
    {
        if (textBox1.Visible==false)
        {
            textBox1.Visible = true;
        }
        else
        {
            textBox1.Visible = false;
        }
    }
//*********************************************************************************************

    private void button4_Click(object sender, EventArgs e)
    {
        OleDbCommand sil = new OleDbCommand("delete * from
kayıtlar",baglanma);
        baglanma.Open();
        sil.ExecuteNonQuery();
        baglanma.Close();
```

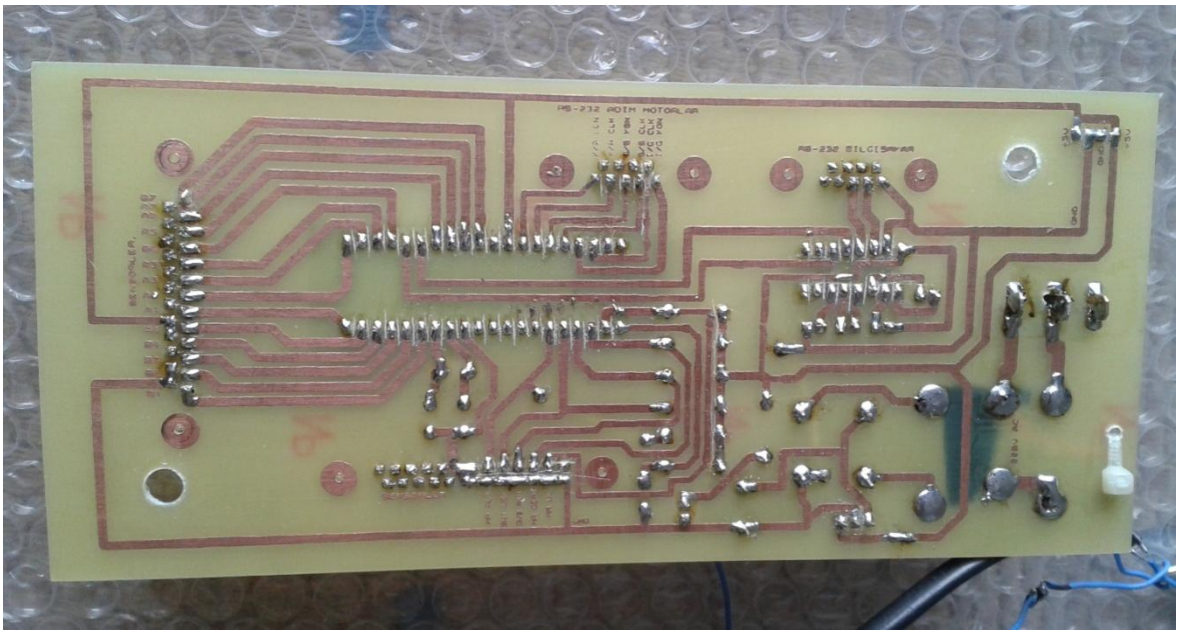
EK-2. (devam) Bilgisayar yazılımı

```
        OleDbCommand komut = new OleDbCommand("Select * From kayıtlar ",
baglanma);
        OleDbDataReader okuyucu = null;
        baglanma.Open();
        okuyucu = komut.ExecuteReader();
        listView1.Items.Clear();
        while (okuyucu.Read())
        {
            ListViewItem Liste = new ListViewItem(okuyucu["id"].ToString());
            Liste.SubItems.Add(okuyucu["plaka"].ToString());
            Liste.SubItems.Add(okuyucu["giriş"].ToString());
            Liste.SubItems.Add(okuyucu["port"].ToString());
            Liste.SubItems.Add(okuyucu["çıkış"].ToString());
            Liste.SubItems.Add(okuyucu["ücret"].ToString());
            Liste.SubItems.Add(okuyucu["kullanım"].ToString());
            listView1.Items.Add(Liste);
        }
        baglanma.Close();
    }

//*********************************************************************************************

    private void Form1_Shown(object sender, EventArgs e)
    {
        MessageBox.Show("Elektromekanik Sistem ile Bağlantı Sağlanmıştır");
    }
      }
}
//*********************************************************************************************
```

EK-3. Elektronik devrenin resimleri

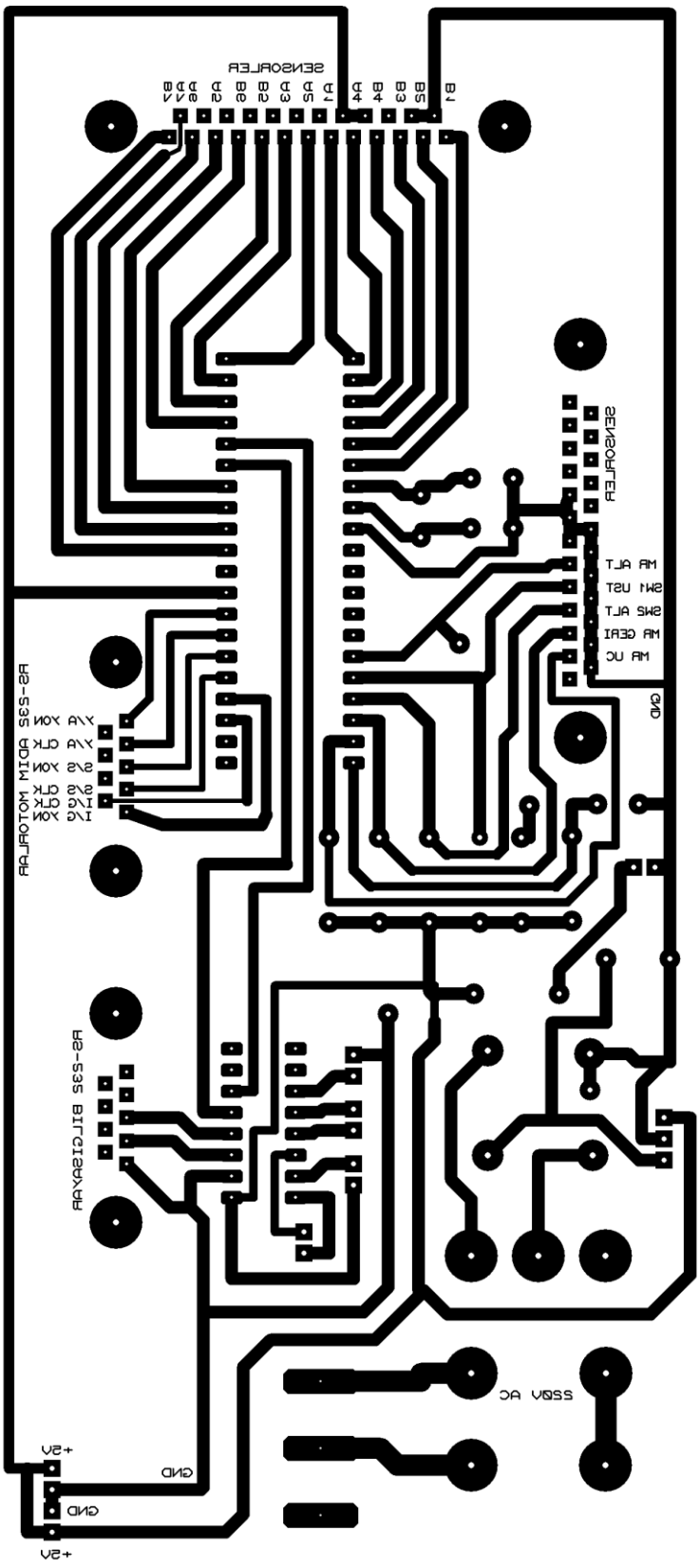


Şekil 3.1. Elektronik devrenin eleman yüzeyinden görünüşü



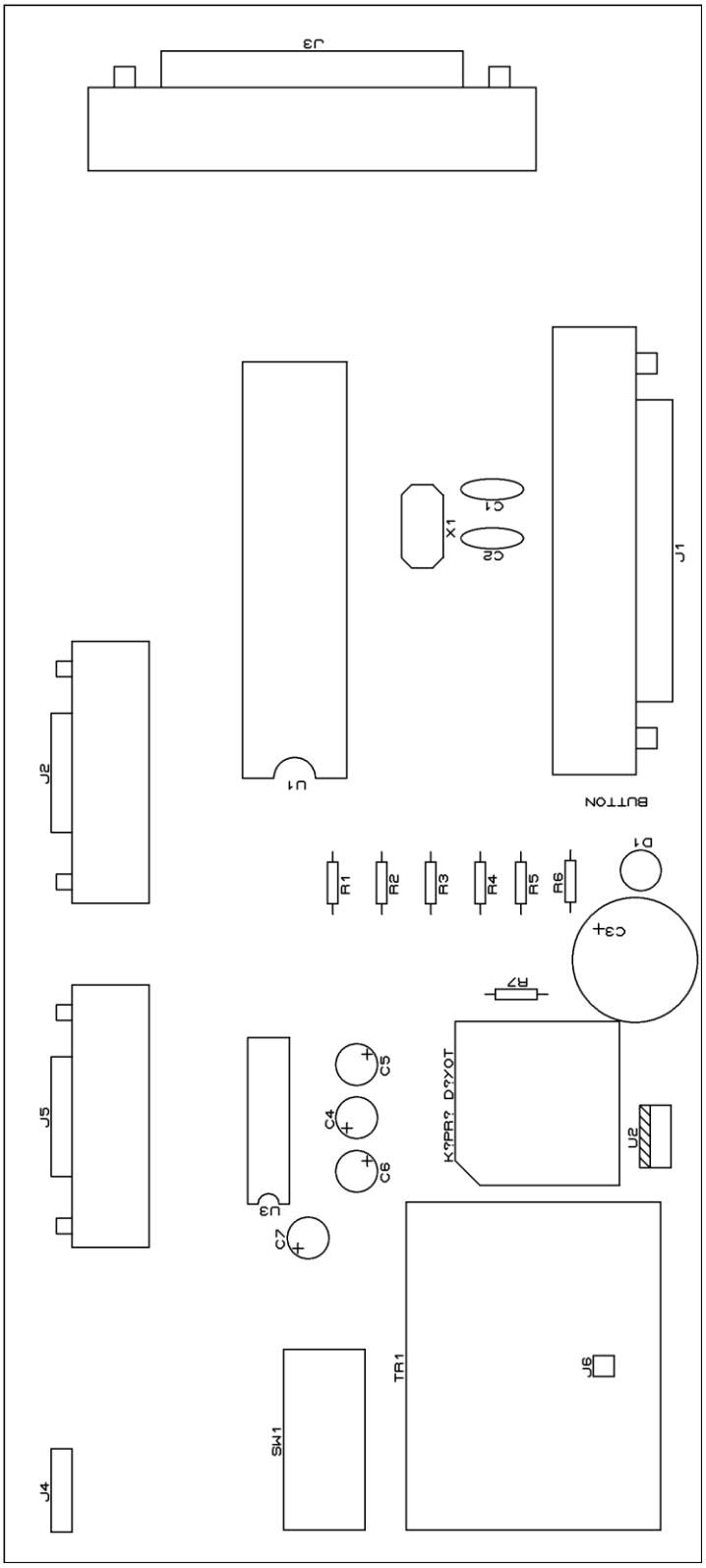
Şekil 3.2. Elektronik devrenin baskı devre yolları yüzeyinden görünüşü

EK-4. Elektronik devrenin baskı devre ve eleman yerleşim planı şemaları



Şekil 4.1. Elektronik kontrol ünitesi baskı devre şeması

EK-4. (devam) Elektronik devrenin baskı devre ve eleman yerleşim planı şemaları



Şekil 4.2. Elektronik kontrol ünitesi eleman yerleşim planı

ÖZGEÇMİŞ

Kişisel Bilgiler

Soyadı, adı	: ERDOĞDU, Ertuğrul
Uyruğu	: T.C.
Doğum tarihi ve yeri	: 04/04/1980, Ankara
Medeni hali	: Evli
Telefon	: 505 248 99 86
e-mail	: ertugrulerdogdu@gmail.com

Eğitim

Derece	Eğitim Birimi	Mezuniyet Yılı
Yüksek lisans	Gazi Üniversitesi / Elektronik Bilgisayar Eğitimi	Devam ediyor
Lisans	Gazi Üniversitesi / Elektronik Öğretmenliği	2003
Lise	Yenimahalle Anadolu Teknik Lisesi / Elektronik Bölümü	1998

İş Deneyimi

Yıl	Yer	Görev
2013 - Halen	Milli Eğitim Bakanlığı	Müdür Yrd.
2003 - 2013	Milli Eğitim Bakanlığı	Öğretmen

Yabancı Dil

İngilizce

GAZİ GELECEKTİR…