T.C.
YILDIZ TEKNİK ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

MOBİL KEŞİF ROBOTUNUN BİLGİSAYAR İLE KONTROLÜ

Elektronik ve Haberleşme Mühendisi Onur ÇELİK

FBE Elektronik ve Haberleşme Mühendisliği Anabilim Dalı
Elektronik Programında Hazırlanan

YÜKSEK LİSANS TEZİ

Tez Danışmanı : Prof. Dr. Herman SEDEF

İSTANBUL, 2010

İÇİNDEKİLER

Sayfa

KISALTMA LİSTESİ

RTP	Real Time Protocol
JMF	Java Media Framework
CNC	Computer Numerical Control
IP	Internet Protocol
Wi-Fi	Wireless Fidelity
RF	Radio Frequency
DC	Direct Current
USB	Universal Serial Bus
TCP	Transmission Control Protocol
JAR	Java Archive
PDA	Personal Digital Assistant
PC	Personal Computer
GSM	Global System for Mobile Communications
GPRS	General Packet Radio Service
EDGE	Enhanced Data Rates for GSM Evolution
VPN	Virtual Private Network
OEM	Original Equipment Manufacturer
API	Application Programming Interface
ATM	Automated Teller Machine

ŞEKİL LİSTESİ

ÇİZELGE LİSTESİ

ÖNSÖZ

Gerek tez çalışmam sırasında gerekse lisans ve yüksek lisans öğrenimim boyunca göstermiş olduğu ilgi, alaka ve hoşgörüden dolayı Sayın Prof. Dr. Herman SEDEF'e, proje boyunca birlikte çalıştığım ve çalışmalar sırasında desteğini hiçbir zaman esirgemeyen Erkan YİĞİTER'e, projenin mekanik tasarımını gerçekleştirmelerinin yanı sıra çalışmalarımız boyunca tecrübe ve fikirlerini paylaşarak projenin gelişimine büyük katkı sağlayan Hasan ŞEREF ile Aykut ŞEREF'e ve beni bugünlere getiren sevgili aileme teşekkürü bir borç bilirim.

Onur ÇELİK

15/02/2010

ÖZET

Bu çalışmada Wi-Fi kullanılarak, üzerine kamera monte edilmiş paletli bir mobil robotun uzaktan klavye yön tuşları kullanılarak manuel ve çizilen sanal bir rota üzerinde gidecek şekilde otomatik olarak kontrolü amaçlanmıştır. Java programlama dili kullanılarak tasarımı gerçekleştirilen kontrol yazılımı sayesinde, robotun internet üzerinden ya da yerel bir ağ içerisinde bilgisayar ya da benzeri java uyumlu mobil cihazlar kullanılarak kontrol edilmesine olanak sağlanmıştır. Ayrıca robot üzerinde bulunan kamera görüntüsü ve ortamdaki ses de kablosuz ağ aracılığıyla kullanıcıya iletilebilmektedir. Bu sayede insanların giremediği ya da girmesinin tehlikeli olduğu alanlarda, robotun yanında olmadan, gözlem ve keşif yapmak mümkün olmaktadır. Kontrol yazılımlarının Java programlama dili ile yazılması sayesinde robot kontrol programının farklı platformlar üzerinde de değişikliğe gerek duymadan çalıştırılabilmesi sağlanmıştır. Bunun sonucu olarak esnek bir yapıya sahip ve geliştirmeye açık bir mobil robot platformu ortaya çıkmıştır.

Anahtar Kelimeler: Mobil keşif robotu, kablosuz kontrol, Wi-Fi, java, internet.

ABSTRACT

In this study, a Wi-Fi based mobile robot control is aimed both manually using arrow keys of keyboard and automatically on a path which is drawn on a virtual map. Using the control software which is developed by Java programming language, robot control is enabled through a local area network or internet via computer or java compatible mobile devices. The video and sound of the environment are captured via a webcam which is mounted to robot and transferred to the user through wireless network. As a result of this, it is possible to perform exploration and observation in the environment which can be dangerous for human life. Since the control softwares are developed by Java, it is enabled to run robot control software on various platform without making any changes. These features introduce a mobile robot platform which is flexible and open to development.

Keywords: Mobile exploration robot, wireless control, Wi-Fi, java, internet.

1. GİRİŞ

Önceleri tüm işlevleri uzaktan bir operatör yardımıyla kontrol edilen mobil robotlar günümüzde insanların yapmasının tehlikeli veya zor olduğu birçok uygulamada kullanılmaya başlanmıştır. Bu uygulamaların çeşitliliğinin artmasıyla birlikte robotların gerçekleştirmesi gereken işlevler ve bu işlevlerin karmaşıklığı da artmış, bunun bir sonucu olarak, sabit mesafeden takip, engel algılama, hız ve pozisyon denetimleri, rota takibi gibi görevlerin yerine getirilebilmesi için kontrol amaçlı bilgisayar kullanımı zorunlu hale gelmiştir.

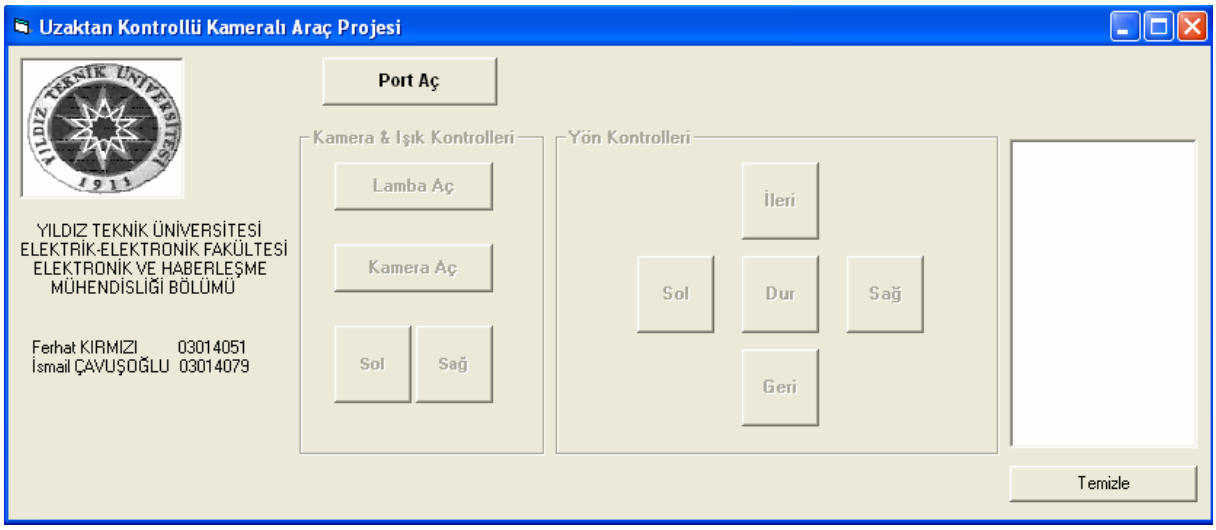
Yarı otomatik olarak adlandırabileceğimiz bu robotlar, karmaşık bir takım işlemleri kendi başlarına gerçekleştirirken bazı işlemler için ise bir operatör kontrolüne ihtiyaç duymaktadır. Bu sayede hem operatörün yükü azalmakta hem de insan hatasına açık bazı görevlerin başarılı olarak yerine getirilme oranı artmaktadır. Yarı otomatik robotların bir ileri adımı ise tam otomatik robotlardır. Bu robotlarda bir operatör mevcut değildir. Sadece yapılması istenen işi yükleyen bir programcı bulunmaktadır. Tam otomatik robotlar, çevrelerini kendileri algılar ve yapacakları hareketlere kendileri karar verirler.

Yarı otomatik robotların kontrolü için kullanılan yazılımlar operatör kontrol programları ve robot kontrol programları olmak üzere işlevsel olarak ikiye ayrılabilir. Robot kontrol programları, robotun kendi başına gerçekleştirmesi gereken görevleri yerine getirirken, operatör kontrol programları ise robotun kendi başına gerçekleştiremediği işlemler için kullanıcıya bir arayüz sunmaktadır. Operatör programlarında görsel tarafı kuvvetli nesne yönelimli programlama dilleri tercih edilirken, robot kontrol programlarında prosedürel diller de kullanım alanına sahip olabilmektedir.

Mobil robotlar ile kullanıcı arasında bir haberleşme ortamına ihtiyaç duyulmaktadır. Bu haberleşme genel olarak kablosuz haberleşme teknikleri ile gerçekleştirilmekte, kullanılacak kablosuz haberleşme tekniği uygulamanın ihtiyaçlarına göre çeşitlilik göstermektedir. Bu tekniklerden en yaygın olanları RF, Bluetooth ve Wi-Fi olmakla beraber gerek sağladığı yeterli bant genişliği ve gerekse güvenilir veri şifreleme yöntemleri ile Wi-Fi, mobil robot çalışmalarında kullanım için oldukça elverişlidir.

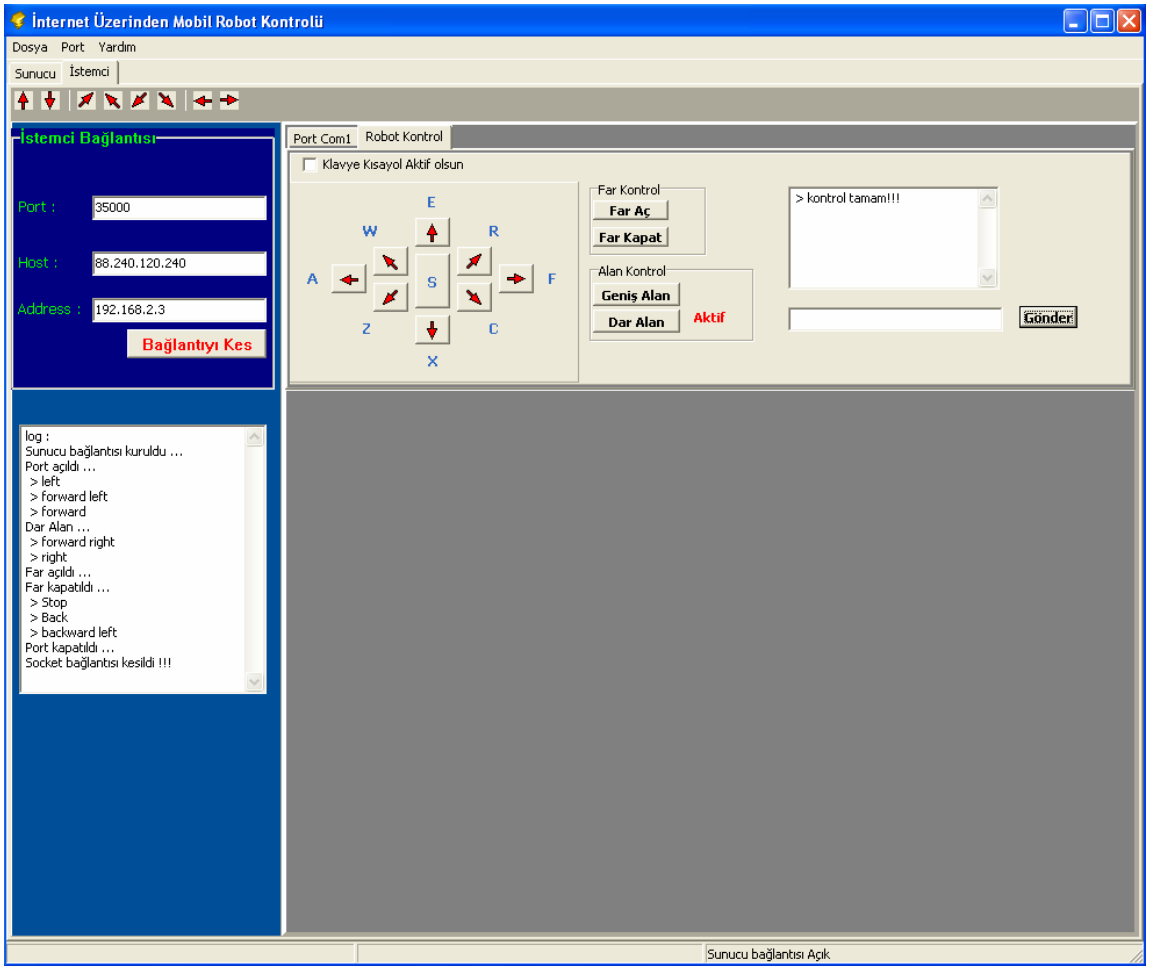
Mobil robotların bilgisayar aracılığıyla kontrolü üzerine birçok çalışma yapılmıştır. Bu çalışmalarda mobil robotun bilgisayar ile kontrol edilebilmesi için farklı programlama dilleri kullanılarak farklı işlevleri yerine getiren kullanıcı arayüzleri tasarlanmıştır. Çavuşoğlu ve Kırmızı (2007), seri port ile haberleşebilen uzaktan kumandalı kameralı bir araç projesi gerçekleştirmişlerdir. Microsoft Visual Basic 6.0 ile hazırlanan ve Şekil 1.1'de ekran

görüntüsü verilen kullanıcı arayüzü yardımıyla alınan kullanıcı direktifleri doğrultusunda bilgisayarın seri portuna aktarılan veriler, RF verici modül aracılığı ile araç üzerinde bulunan RF alıcı modüle kablosuz bir şekilde iletilerek haberleşme sağlanmaktadır. Araca monte edilmiş kamera yardımıyla bilgisayar başındaki kullanıcı kumanda ettiği aracın hangi yöne gittiğini görebilmektedir. Haberleşme yöntemi olarak RF tercih edildiğinden, araç ile haberleşme mesafesi RF alıcı-verici modül çiftinin kapsama alanı ile yakından alakalıdır. Devre üzerinde bulunan RF, kapalı alanlarda 50 metre, açık alanda ise 100 metre mesafe içerisinde çalışmaktadır.



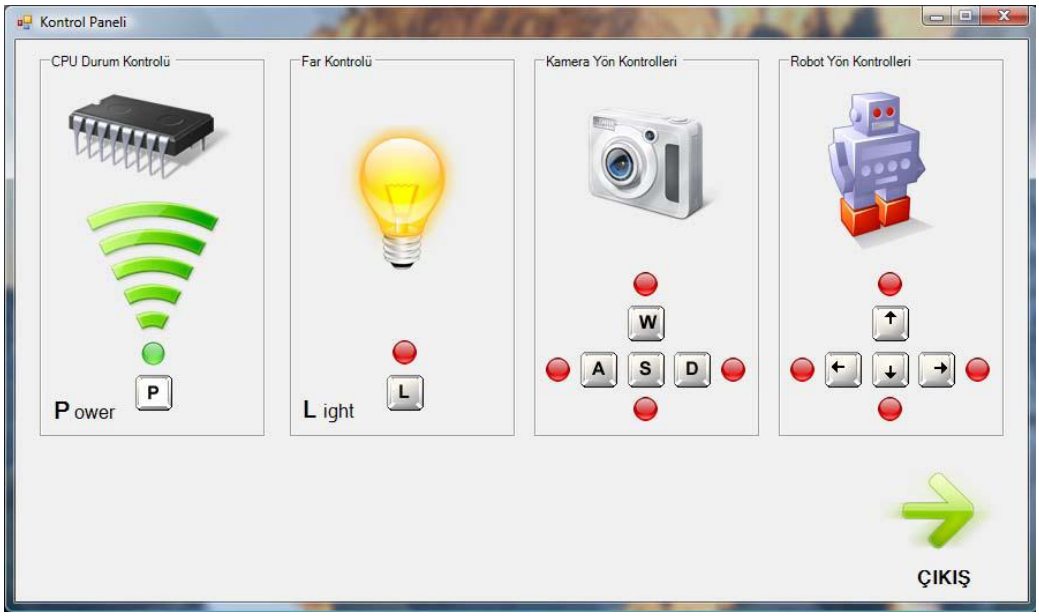
Şekil 1.1 Uzaktan kontrollü kameralı araç projesi kontrol yazılımı

Ünlü (2007), bilgisayarla RF aracılığıyla haberleşen DC motorlu mobil bir robotun, internet üzerinden kontrolünü gerçekleştirmiştir. Bilgisayara yüklü kontrol programı ile robotun hareketleri kontrol edilmiştir. Sistem üç ana kısımdan oluşmaktadır. Bunlar, internet üzerinden haberleşmeyi sağlayan kontrol programı, bilgisayar tarafı haberleşme kartı ve mobil robot tarafı haberleşme ve kontrol kartıdır. C++ kullanılarak geliştirilen kullanıcı program, sunucu taraftaki mobil robotu kontrol edebilmek için internet ağında herhangi bir yerde çalıştırılmalıdır. Görevi, sunucu tarafla internet bağlantısını kurmak ve mobil robotun kontrolü için gerekli ayarların karşı tarafla uyumlu olarak yapılmasını gerçekleştirmektir. Bu programın ekran görüntüsü Şekil 1.2'de verilmiştir. Sunucu taraftaki program ise mobil robotun bulunduğu bilgisayar tarafında çalışmaktadır ve istemci bilgisayarın, sunucu tarafındaki bilgisayara internet üzerinden bağlantı yapılabilmesini ve seri porta ulaşmasını sağlamaktadır. Bu projede de haberleşme yöntemi olarak RF tercih edilmiştir.



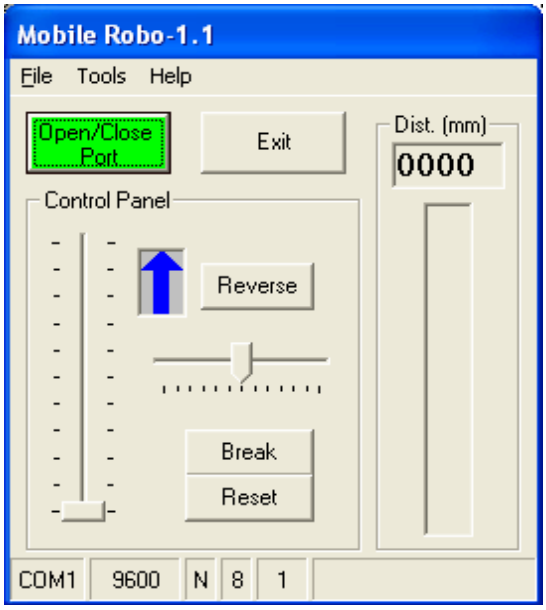
Şekil 1.2 İnternet üzerinden mobil robot kontrolü

Yalvaç (2008), bilgisayar aracılığı ile kontrol edilebilen mobil bir robot projesi gerçekleştirmiştir. Bu projede, bilgisayarla RF aracılığı ile haberleşen DC ve servo motorlara sahip mobil bir robotun kontrolü amaçlanmıştır. Şekil 1.3'te ekran görünümü verilen ve Visual Studio 2008 Basic .Net'de geliştirilmiş olan yazılım sayesinde, robotun ileri-geri, sağ-sol hareketleri ile robot üzerinde bulunan robot kola ait kameranın yine yukarı-aşağı, sağ-sol hareketleri ve farların açık-kapalı durumları kontrol edilebilmektedir. Yine yazılım sayesinde robotun bulunduğu ortam görüntüsünün fotoğrafi veya sesli olarak video kaydı çekilebilmektedir.



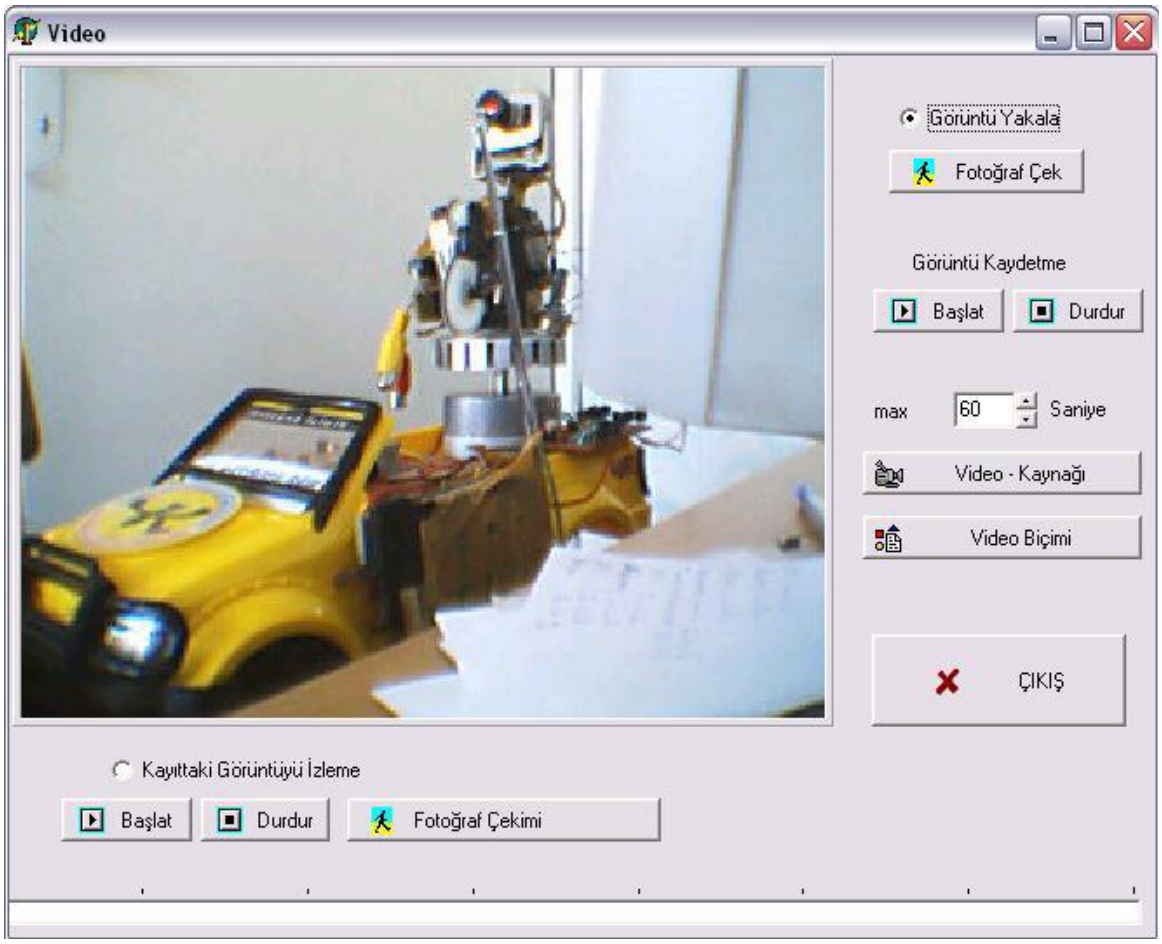
Şekil 1.3 Mobil araştırma robotu kontrol paneli

Yıldız (2004), bir gezgin robot sisteminin donanımının, temel hareket yazılımının ve kullanıcı arayüzünün fiziksel olarak gerçeklenmesini sağlamıştır. Tasarlamış olduğu denetim yazılımı ile hız, motor dönme yönü, fren ve ön tekerlek açısı kontrolü, robotun önündeki engele olan mesafesinin okunması ile başlangıç değerine ilk koşullama gibi işlemler gerçekleştirilmiştir. Şekil 1.4'te bu projeye ait kontrol programının ekran görüntüsü verilmiştir. Kişisel bilgisayar üzerinden denetimi sağlayan ara yüz ile gezgin robot arasındaki haberleşme RS232 standardında asenkron olarak gerçeklenmiştir.



Şekil 1.4 Araba benzeri bir gezgin robotun kontrol programı

Çayıroğlu ve Şimşir (2008), RF sinyalleri ile uzaktan kumanda edilen bir robot araba üzerine yine kontrolü ve görüntü alımı uzaktan olan bir kamera bağlanarak kontrol işlemi gerçekleştirmiştir. Uzaktan gönderilen sinyallerle kamera düşey ve yatay eksende step motor kontrolü ile çevrilmiş ve alınan görüntüler gerçek zamanlı olarak bilgisayar ekranından izlenmiştir. Tüm kontrolleri bilgisayardan yapabilmek için Delphi 7.0 yazılımı kullanılarak Şekil 1.5'te görülen arayüz kontrol programı yazılmıştır. Program görüntü izleme ve kontrol amaçlı olarak kullanılmıştır. Görüntü ile ilgili olarak görüntü yakalama, fotoğraf çekme, video kayıt, video ayarları, video kaydı izleme gibi işlemle yapılabilmektedir. Bu işlemler için Delphi 7.0'ın ComCtrls, VideoCap, VideoMci gibi özel kütüphaneleri ve bu kütüphanelere ait komutları kullanılmıştır. Kontrol amaçlı olarak ise arabanın kontrolü, adım motorun kontrolü, DC motorun kontrolü gibi işlemler gerçekleştirilmiştir. Bu cihazların tamamının kontrolü bilgisayarın paralel portu aracılığıyla gerçekleştirilmiştir. Bu maksatla Delphi7.0'ın ComCtrls, Controls kütüphaneleri ve kütüphanelere ait komutları kullanılmıştır.



Şekil 1.5 Mobil aracın arayüz kontrol programı

Bu çalışmada ise mobil bir robotun Java programlama dili kullanılarak tasarlanan ve sunucu-

istemci mimarisine dayalı programlar ile günümüz popüler kablosuz haberleşme teknolojilerinden Wi-Fi üzerinden kontrolü amaçlanmıştır. Projede, gerek platform bağımsız olması, gerek IP tabanlı gerçek zamanlı veri alışverişi uygulamalarında sıklıkla kullanılıyor olması ve gerekse ücretsiz derleyici ve hazır kütüphanelere sahip olması nedeniyle java programlama dili tercih edilmiştir.

Mobil robot üzerinde konumlandırılmış kameradan alınan görüntü ve sesin kullanıcıya aktarılması işlemleri yine Java'nın hazır sınıfları içerisinde sunmuş olduğu Java Media Framework'ten yararlanılarak gerçek zamanlı veri iletişim protokolü RTP üzerinden gerçekleştirilmiştir. Robotun kontrolü, hem bir operatör tarafından klavyedeki yön tuşları yardımıyla hem de pratikteki uygulamalardan farklı olarak operatörün kontrol programı üzerinde çizdiği sanal rotanın takibi şeklinde yapılabilmektedir. Bu kapsamda takip edilecek fiziksel bir girdiye ihtiyaç duymadan sanal bir rota üzerinden kontrolün sağlanıyor olması, projenin benzer uygulamalarla arasındaki farkı öne çıkarmaktadır.

Robota erişim, bir yerel ağ ya da internet üzerinden kullanıcı kontrol programı aracılığıyla parola korumalı olarak sağlanmakta, robotun kontrolü ancak belirlenen parolaya sahip operatörler tarafından gerçekleştirilebilmektedir. Kontrol amaçlı gerçekleştirilen tasarım, yazılım mantığı ve platform bağımsız olması da göz önüne alındığında, bu konuda yapılmış olan çalışmalardan farklı olarak, yapılacak değişikliklere açık olup, istenildiği takdirde farklı uygulamaların, farklı sistemler üzerinde kolaylıkla gerçekleştirilip adapte edilebileceği temel bir yapı sağlamaktadır.

2. ROBOTLAR

Bu bölümde, robotlar, robotların tarihsel gelişimi, tercih edilme nedenleri, mobil robotlar ve robot programlama yöntemleri hakkında genel bir bilgi verilmektedir.

2.1 Robotun Tanımı

Robot, bu konuda çalışmalarıyla tanınan Maja Mataric'in yaptığı tanıma göre, ortamdan topladığı verileri sahip olduğu bilgiyle sentezleyerek, anlamlı ve amaçlarına yönelik bir şekilde hareket edebilen ve bunu güvenli bir biçimde yapabilen bir makinedir. Amerikan Robot Enstitüsü tarafından ise robot şu şekilde tanımlanmaktadır: "Robot, yeniden programlanabilen, maddeleri, parçaları, aletleri, özel cihazları yerinden oynatabilen çok fonksiyonlu bir makinedir." Sanayi robotunun en kapsamlı tanımı ve robot tiplerinin sınıflandırılması ise ISO 8373 standardında belirlenmiştir. Bu standarda göre bir robot şöyle tanımlanır: "Endüstriyel uygulamalarda kullanılan, üç veya daha fazla programlanabilir ekseni olan, otomatik kontrollü, yeniden programlanabilir, çok amaçlı, bir yerde sabit duran veya hareket edebilen manipülatördür."

Bir makinenin robot olarak tanımlanabilmesi için bulunduğu ortamı denetleyecek algılayıcılara, bu algılayıcılardan gelen bilgiyi işleyip sonuç çıkaracak işlem birimlerine ve işlem sonuçlarını çıkışa aktarabilecek hareket mekanizmalarına sahip olması gerekir.

Robotlar doğrudan bir operatörün kontrolünde çalışabildikleri gibi bağımsız olarak bir bilgisayar programının kontrolünde de çalışabilirler. Robot denilince insan benzeri makineler akla gelse de robotların çok azı insana benzer.

Günümüzde robotların en büyük kullanım alanı endüstriyel üretimdir. Özellikle otomotiv endüstrisinde çok sayıda robot kullanılmaktadır. Bunların çoğu kol şeklindeki robotlardır. Bunlar parçaları monte eden, birleştiren, kaynak ve boya yapan robotlardır.

Evlerde de robot kullanımı giderek artmaktadır. Başta ABD'de olmak üzere ev işlerine yardımcı olan robotların kullanımı giderek yaygınlaşmaktadır. Yerleri kendi kendine süpüren robot elektrik süpürgeleri büyük talep görmektedir.

2.2 Robotların Tarihi Gelişimi

Robot kelimesi, ilk defa, Karel Capek isimli bir Çek yazarın 1920 yılında yazdığı "Rossum'un Evrensel Robotları" isimli oyununda kullanılmıştır. Yazarın ana dilinde bu kelime, köle anlamına gelmektedir. Oyunda robotlar, Rossum ve oğlu tarafından insanlara hizmet etmek

için yaratılmıştır. Daha sonra 1940'larda Isaac Asimov robotlarla ilgili roman ve hikayeleriyle teknolojiye ışık tutmuş, hatta robotların davranışları ile ilgili aşağıdaki kanunları önermiştir.

1) Robot, hiçbir zaman insana zarar verecek hareketler yapmamalı, ancak, insanın zarar görebileceği durumlarda hareketsiz kalmamalıdır.

2) Birinci kanunu çiğnememek şartı ile robot her zaman insana itaat etmelidir.

3) Birinci veya ikinci kanunları çiğnememek şartı ile robot kendini korumalıdır.

Robotların gelişmesindeki dönüm noktalarını aşağıdaki şekilde özetlemek mümkündür.

- 1801: Programlanabilir dokuma tezgahı
- 1830: Otomatik çıkrık
- 1893: Ayakları ile yürüyen araç
- 1945: Radyoaktif malzemeyi tutmak için teleoperatör
- 1953: Servo denetimli freze tezgahı
- 1954: İlk programlanabilir endüstriyel robot
- 1959: İlk ticari robot
- 1974: Mini bilgisayar kullanan ilk ticari endüstriyel robot

Son onbeş yılda verimliliği arttırmak için yoğun çalışmalar yapılmış ve değişik otomasyon yöntemleri geliştirilmiştir. Bu yöntemlerden birisi de programlanabilir otomasyon yöntemidir. Günümüzde robotlar bu tip uygulamalar için kullanılmaktadır. Örnek olarak kaynak, boyama veya döküm işleri verilebilir. Bu tür bir otomasyonda robotların bilgisayar programları değiştirilerek, değişik işleri yapabilmeleri sağlanabilmektedir.

2.3 Robotların Tercih Edilme Nedenleri

Günümüzde endüstriyel alandaki robotlar, insan işçilerin sıkıcı, kötü, tehlikeli veya çok ince görülmesi gereken işlerde yerini almak üzere tasarlanmaktadırlar. Yaptıkları hareketlerde insan hareketlerini taklit etmektedirler, fakat insanın düşünce ve hareket tarzını kopyalayamazlar. Bazen insanların sahip olmadıkları metotlar ve özellikler yardımıyla bir insanın gösterebileceği performansın üzerine çıkarlar. Örneğin bir robot kızılötesi ve ultrasonik algılayıcılar kullanabilir ve görüşü insandan daha iyi düzeyde olabilir. Göz olarak

kullandığı algılayıcıları insandakinden farklı yerlerde, örneğin parmağında olabilir, böylelikle insanın göremediği şeyleri görebilir ve daha titiz bir çalışma yapabilir. İnsanların çalışamadıkları koşullar altında, elektromanyetik özelliklere, özel tutma-kavrama cihazlarına ve değinilen tüm diğer özelliklere sahip bir robot rahatlıkla çalışabilir, çoğu zaman da daha ucuza daha iyi performans gösterebilir.

Aşağıda verilenler robotun insan yerine tercih edilme sebeplerinden bazılarına gösterge olmaktadır.

- Emek maliyetlerinin yüksek olduğu ülkelerde robot sayısının artmasındaki en önemli unsurlardan biri üretim maliyetlerinin robot kullanımıyla düşürülmesidir. Sosyal, sağlık ve emeklilik gibi yardımların da göz önüne alınmasıyla ortaya çıkan maliyet robot için harcanan paranın üç-dört katını bulabilmektedir. Ancak, emeğin ucuz olduğu ülkelerde tam tersi bir durum ortaya çıkmaktadır.

- İşçilerin görevde olduğu sürecin % 15-20'lik bir bölümü ortaya çıkan yorgunluğun giderilmesi ve diğer ihtiyaçların karşılanması için geçer. Bu süre robotlarda %2'yi geçmemektedir.

- Robotlarda yorgunluk ve dikkat kaybı söz konusu olmadığından hatalı imalat sayısı insanın neden olduğu hatalı imalat sayısına göre neredeyse sıfırdır. Böylece hatalı imalatın üretim maliyetindeki payı çok düşük kalır.

- Robotların insanlar gibi haftalık 40 saat çalışma süresi kısıtı yoktur. Tüm hafta gece gündüz çalışabilirler. Dolayısıyla insanlar bir işte vardiyalar halinde çalışıp sürekli değişim olurken o işte aynı robot devamlı çalışmaktadır. Bu yönüyle de birim üretim maliyeti çok düşmektedir.

- Robotlar bazı işlerde insanlara kıyasla çok daha hızlı çalışırlar. Örneğin bir ark kaynağı robotu dakikada 75 cm. kaynak yapabilirken ortalama bir kaynak ustası dakikada ancak 25 cm. kaynak yapabilir. Hacimlerin ve ekipmanların daha etkin kullanımı ile belirli bir program dahilinde üretkenlik artmaktadır.

- Robotların pozisyonlama yeteneği insana göre daha yüksektir. Robot ile gerçekleştirilen bir kaynak dikişi genellikle taşlanmaya ihtiyaç duymaz ve robot ile üretilen parçalar insanın ürettiklerinden daha iyi toleranslara sahiptirler. Bazen operasyon hızının yüksek olması kaliteyi arttırabilir. Örneğin ince parçaların kaynağının hızlı yapılması ısı yayılımını önleyerek parçalardaki çarpılmaların

azalmasını sağlayacaktır. Ayrıca hızın kontrol edilmesiyle homojen bir kaynak dikişi elde edilecektir.

- Sıcak dövme preslerinde tezgah yüklenmesi ve boşaltılması robotların ilk uygulama alanlarından biri olmuştur. Yüksek sıcaklıktaki dövme esnasında parça belirli bir konumda tutulmalıdır. Önceleri bu işi, iki kişi uzun maşalar vasıtasıyla yaparken robot uygulamasıyla tutma işi tutucu yardımıyla robot tarafından yapılmaya başlanmıştır. Böylece daha büyük hızlara ulaşılırken çalışanlar da sıcak parça ve kıvılcım sıçrama tehlikesinden uzaklaştırılmış olurlar. Daha iyi bir konumlamayla da ürün kalitesi arttırılmış olur.

- Bazı boyama işlerinde asit boyalar kullanılır ve bu boyalar da boyama görevlisinin sağlığı açısından çok tehlikelidir. Personelin sızdırmaz giysiler ve başlıklarla çalışmaları ve takılan başlıklara sürekli hava beslemesi yapılması gerekir. Bu koşullar altında çalışmak verimsiz ve yorucudur. Oysa aynı iş, daha önceden programlanmış hareketleri yapan bir robot vasıtasıyla daha hızlı olarak daha yüksek kalitede gerçekleştirilebilir.

- Her idarecinin birbirleriyle sürekli rekabet eden ve her söyleneni yapan çalışanları tercih edeceği aşikardır. Aslında böyle bir durum her idarecinin hayalidir. Bazıları bir robot sistemi oluşturarak ve diğer ekipmanları da bu sisteme uydurarak bu hayali gerçekleştirmeye başlamışlardır. Yapılacak işlemler çok hassas biçimde programlanabilir ve malzemeler robot iş hücrelerine bilgisayar kontrolü altında ulaştırılabilir.

- Robotlar, önceden programlanmış hareketleri büyük bir doğrulukla gerçekleştirdikleri gibi ne yapıldığını da büyük bir doğrulukla kaydedebilirler. Bu kayıtlar programlama, planlama ve kontrol işlemlerinin iyileştirilmesinde önemli bilgileri teşkil eder.

- Robotlar, yeniden programlanma hatalarının düzeltilmesi işleminin basitliği dolayısıyla değişik işlere adaptasyonda önemli güçlük doğurmazlar ve işlemler uzun süreli üretim durmasına neden olmaz. Oysa sabit otomasyonda değişiklik yapmak, uzun süreli üretim aksamalarına neden olmaktadır.

- Çeşitli uygulama durumlarında yeniden programlanabilme yeteneği, tutucunun değiştirilebilme özelliği, sistem ömrünü uzatır. Sabit otomasyon sistemlerinde değişiklik yapmak önemli bir harcamayı gerektirir ki bu harcama zaman zaman

sistemin yeniden oluşturulma maliyetine yakın olabilir.

Robot işgücü ve insan arasında tercih yapılırken maliyetler de mutlak olarak göz önünde bulundurulmalıdır. Sonuçta robotların da makineden bir farkı yoktur. Çalışma alanının maksimum kullanımı, robot parkını kurmak için gerekli yatırım ve diğer maliyetler işletme açısından hayati önem taşırlar. Yatırımın geri dönüş süresi de göz ardı edilmemesi gereken bir konudur.

2.4 Robot Programlama Yöntemleri

Robot sistemlere program yazılarak aynı robot sistemin farklı zamanlarda değişik işler yapması sağlanabilir. Program bir öğretim paketidir. Robot sisteme ait her bir hareket elemanının ne zaman ve ne kadar hareket edeceği program içerisinde belirtilir. Çok amaçlı robotlarda, genel sistem kontrol elemanı bilgisayardır. Robotların kullanım alanlarının genişlemesi ve yazılım teknolojisinin gelişmesi, belli bir amaç için yapılan robotların, o amaç doğrultusunda programlanması ihtiyacını da beraberinde getirmiştir. Bu nedenle, değişik firmalar, ürettikleri robotlar için farklı yazılımlar geliştirmiş ve kullanmışlardır. Günümüzde robotlar için yapılan yazılımlar artık ayrı birer ürün değil, robotların birer parçası haline gelmişlerdir.

Robot yazılımı üreten şirketler öncelikle simülasyonlar üzerinde çalışırlar. Buna off-line programlama denir. Robot üretilmeden önce, yazılım aşamasında, hareketleri, hangi şartlarda ne tür tepkiler vereceği, öğrenme kabiliyeti gibi özellikleri önceden bilgisayar ortamında off-line programlama tekniği ile test edilir.

Robot yazılımlarında programın çalışma süresi genellikle öncelikli öneme sahip değildir. Robot programcıları daha çok ürettikleri programın robota minimum sayıda yükleme gerektirmesini amaçlarlar. Bu sebeple, off-line programlama aşamasında üretilen programın tam olarak isteneni yapabilmesi hususu ele alınır.

Off-line robot programlama ve simülasyon, robot modellerinin simülasyonunun mümkün olduğu durumlarda, robot programlarının yapımına yardımcı olmaktadır. Robot programları, gerçek robota yüklenip çalıştırılmadan önce simülatörler aracılığıyla test edilirler. Bunun esaslı amaçlarından biri de programdaki olası hatalar yüzünden robota ya da çevredekilere zarar gelmesini engellemektir. Ayrıca off-line programlama robot teknolojisi kullanan birçok fabrikada önemli ölçüde maddi kazanç sağlamaktadır. Çünkü çalışma sırasında meydana gelebilecek hatalar da, büyük robotlar yerlerinden çıkarılmadan, simülasyon yardımıyla

görülebilmekte ve düzeltilebilmektedir.

Robot programlaması ve simülasyonunda karşılaşılan en büyük problemlerden biri robotun hareket eden kısımlarının çevreye ve robotun bağlantılarına göre nasıl yönlendirileceğidir. İyi bir robot simülasyon programı, gerçek robotun yapabileceği her şeyi simüle edebilmelidir.

2.4.1 Robot Programlama Dilleri

Robotların programlanmasında birçok farklı programlama dili mevcuttur. Bunlardan önemli olan birkaçı hakkında bu bölümde genel bilgi verilmiştir.

Robotscript, Evrensel Robot Denetimcisi (Universal Robot Controller) tarafından kontrol edilen bütün robot modellerini sanal olarak programlamak için kullanılır. On-line olarak Windows NT istasyonlarında ya da off-line olarak Windows yüklü herhangi bir PC'de kullanılabilir. Robotscript, grafiksel kullanıcı arabirimi (GUI) ile kullanıcıya kolaylık sağlar ve hata olasılığını azaltır. Ayrıca yazılım geliştirme paketi yardımıyla her türlü kullanıcı için farklı arayüzler tasarlanabilir.

Robotscript, makro benzeri bir robot programlama dilidir ve ActiveX programıyla beraber çalışır. Microsoft Visual Basic programı kullanılarak geliştirildiği için, Microsoft C++ ya da Microsoft Office programlarıyla beraber kullanılabilir. Windows ortamına uyumlu olması, programcıya kendi yazılımlarını geliştirirken, robota bilgi yüklerken veya alırken önemli kolaylıklar sağlar.

Robotscript program paketi 4 parça yazılımdan oluşmaktadır: Robotscript ActiveX Kontrolü, ERD kontrol ekranı, dosya yaratıcısı ve yazılım denetimcisi. ActiveX parçası Robotscript yazı dosyasını robot hareketlerine çevirir. İkili (Binary) bilgi dosyası kullanılmadığı için yazılan programları çalıştırmak için derlemeye gerek yoktur. Kontrol ekranı kullanıcının programları çalıştırmasını sağlar. Yazılım geliştirme paketi sayesinde son kullanıcı kendi amaçlarına uygun arayüzler geliştirebilir.

ARAC (Aritmetik Robot Uygulama Denetimi) yazılımı birçok esnek üretim sisteminin çekirdeğini oluşturur. Sistem, basit off-line programlama istasyonu olarak kullanılabilir ve genelde ilk seferde doğru çalışan ve ekstra ayar gerektirmeyen programlar üretilir. Programlamanın yanında, işlem parametrelerini değiştirerek işlemleri on-line olarak birleştirmekte ya da kesmekte de kullanılabilir.

Uygun bir dizi işlemci yardımıyla, ARAC sistemi tamamen otomatik çalışır. Kullanıcı girişi

gerektirmeksizin yeni robot programları üretip bunları bağlı olan robot sistemleri üzerinde çalıştırabilir. Ayrıca eklenebilen ekipmanları sayesinde yardımcı robot elemanlarını da kontrol edebilir.

AML, PASCAL benzeri yüksek seviye bir programlama dilidir. Temel söz dizimi diğer yüksek seviye programlama dillerine benzemektedir. Program, PASCAL'daki gibi "Begin" ve "End" komutları arasına yazılır. Ancak AML, nesneye yönelik (object oriented) bir programlama dilidir. Nesne, bilgi üyeleri ve metotlardan oluşan program bloklarıdır. AML, prensip olarak son kullanıcılara yeni bir nesne tanımlama izni vermez. Bunun temel amacı tecrübesiz son kullanıcıların da kolaylıkla ve güvenle denetim programları yazabilmeleridir.

Bazı robot programlama dilleri internet aracılığıyla insan-robot etkileşimini sağlamayı amaçlamaktadır. Ancak burada eşzamanlı iletişim için gerekli olan yüksek kaliteli bağlantı, alıcıdaki (client) programın uyumluluğu gibi bazı sorunlar çıkmaktadır. Bunun için XML tabanlı bir dil olan RoboML kullanılmaktadır.

2.5 Mobil Robotlar

Robotları, mobil ve mobil olmayan robotlar şeklinde iki kısımda incelemek gerekirse şu tanımlamaları yapmak uygun olacaktır. Bir robotun ulaşabileceği maksimum noktalar kümesinden oluşan yüzeyin hacmine çalışma hacmi denir. Eğer bir robotun çalışma hacmi bir referans koordinat sistemine göre yer değiştirmiyorsa bu robota mobil olmayan robot, yer değiştiriyorsa bu robota mobil robot denilebilir.

Mobil robotlar engellere çarpmadan kendi başına hareket edebilen verilen göreve uygun davranışlara karar verebilen hareketli robot sistemleridir (Yazıcı vd., 2006).

Mobil robotlar hiyerarşik, tepkin ve melez olarak hareketlerini kontrol edebilirler. Hiyerarşik olarak; önceden çevre algılanır, sonra robot planlama yapar ve en son olarak harekete geçer. Tepkin olarak ise robot çevreden aldığı bilgiye göre hareket eder veya tepki verir. Melez kontrolde ise hiyerarşik ve tepkin hareketlerin bileşimine göre davranış gösterir (Murphy, 2000).

Günümüzde, kablosuz haberleşme cihazlarının büyük bir hızla artmasıyla mobil cihazların otomatik kontrolü buna bağlı olarak da geniş uygulama alanlarına sahip olmaları nedeniyle, mobil robotlar üzerine yapılan çalışmalar artarak yaygınlaşmaktadır.

Önceleri mobil robotların tüm aktiviteleri, bir insan operatör yardımıyla uzaktan

gerçekleştirilse de, telerobot olarak adlandırılan mobil robotlar, halen bomba imha robotu olarak ve insan sağlığına tehlikeli ortamlarda taşıma işlerinde kullanılmaktadır. Kullanılan mobil robotun işlevleri arttıkça, tasarımı ve kullanımı da zorlaşmaktadır. Bu amaçla, bazı robotik aktivitelerin (örneğin, sabit mesafeden takip, engel algılama, hız ve pozisyon denetimleri vs.) bilgisayar yardımıyla gerçekleştirilmesi bir zorunluluk olmaktadır. Yarı-otonom robot olarak adlandırılan bu tip robotlar, tehlike arz etmeyen bazı işlemleri kendi başlarına yapmakta veya birden çok alt seviye hareketten oluşan hareket serilerini başarı ile gerçekleştirmektedirler. Bu sayede operatörün yükü azalmaktadır.

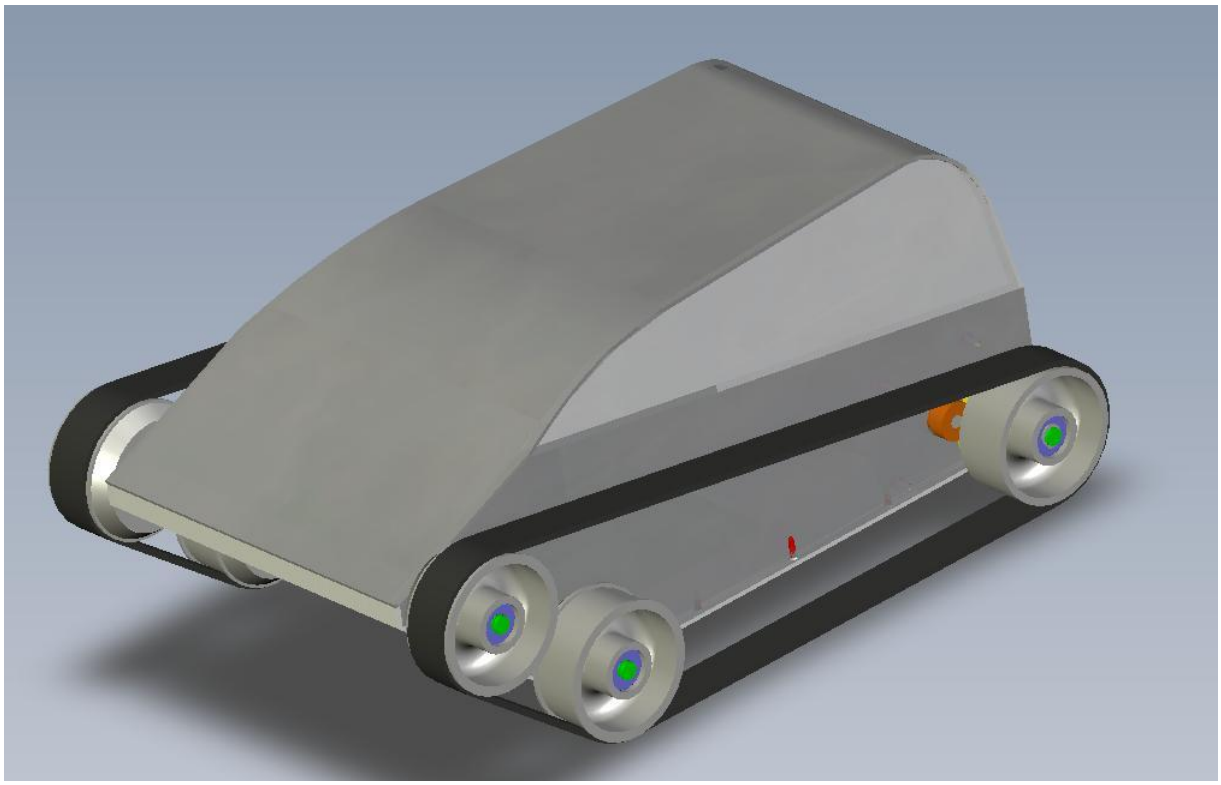
2.5.1 Mobil Robotların Kullanım Alanları

Robotların bir alt türü olan "Mobil Robotlar" kısıtlı kullanım alanına sahip olsalar da serbest taban hareketine ihtiyaç duyulduğu halde insan sağlığına zararlı ortamlarda veya insanların kolay çalışamayacağı ölçüde küçük alanlarda kullanımları yaygınlaşmaktadır. Bu amaçla geliştirilen mobil robotlar, fabrika otomasyonundan uzay araştırmalarına, nükleer atık toplama gibi insanlar için zararlı olan ya da gezegenlerin keşfi gibi kabiliyetlerini aşan işlerde ve de engellilere refakat etme gibi destek amaçlı olarak çok geniş bir kullanım alanına sahiptir.

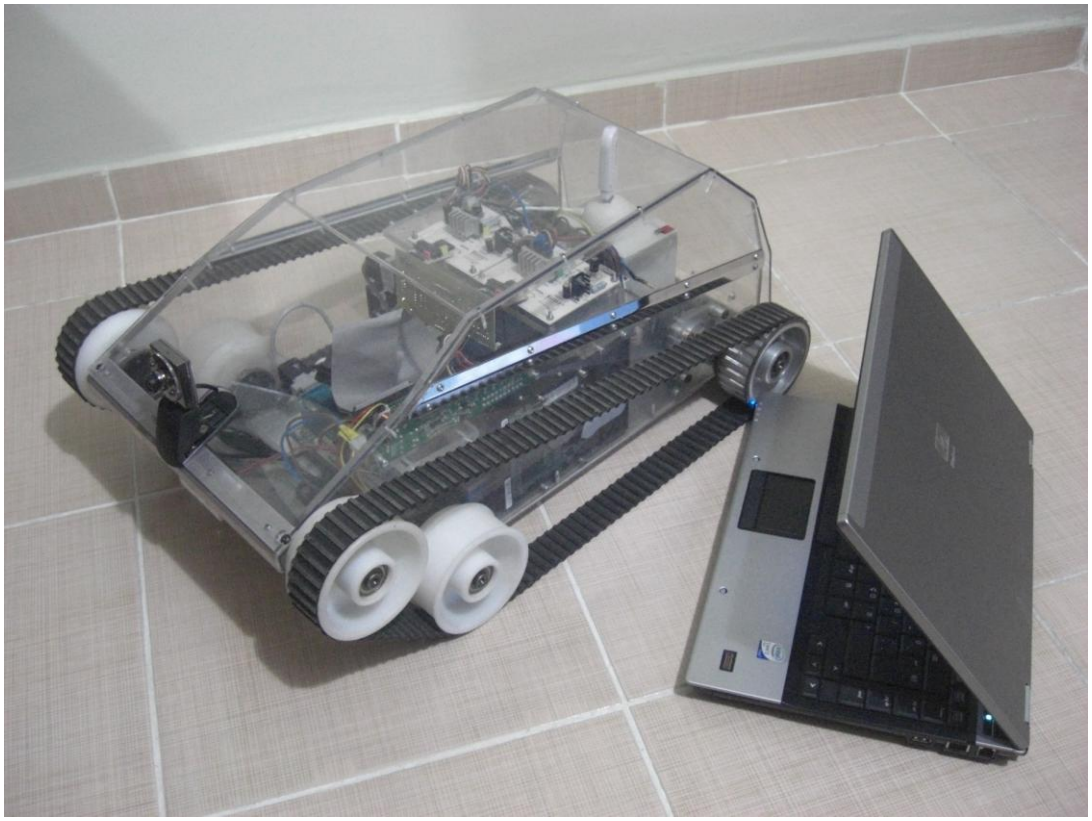
Mobil robotların diğer kullanım alanlarına örnek olarak, toksit atıkların temizlenmesi, nükleer atıkların temizlenmesi, patlayıcıların imhası, biyolojik atıkların taşınması, karantina altındaki ortamlarda servis robotu olarak, yüksek binaların dış camlarının temizliği, madenin yerinin tespit edilip çıkartılması ve taşınması, gezegenlerin keşfi ve incelemesi, uzay istasyonlarının yapımı, deniz altında batık arama ve kurtarma, engelli insanlar için refakatçi olarak, depremlerde insan kurtarma görevleri, fabrika üretim bantlarında parça taşıma verilebilir.

3. MOBİL KEŞİF ROBOTU

Bu çalışmada kablosuz ağlar üzerinden kontrol edilebilen, insanların girmesinin tehlikeli olabileceği veya mümkün olmadığı alanlarda gözlem ve inceleme yapma amaçlı olarak kullanılabilecek Şekil 3.1'de üç boyutlu görüntüsü verilen bir mobil keşif robotu tasarlanmıştır. Bu robot, kablosuz bir ağ üzerinden kontrol edilebildiği için, kablosuz ağın internet çıkışının olması durumunda internet üzerinden de kontrol edilebilmektedir.



Şekil 3.1 Mobil keşif robotunun üç boyutlu dış görünümü



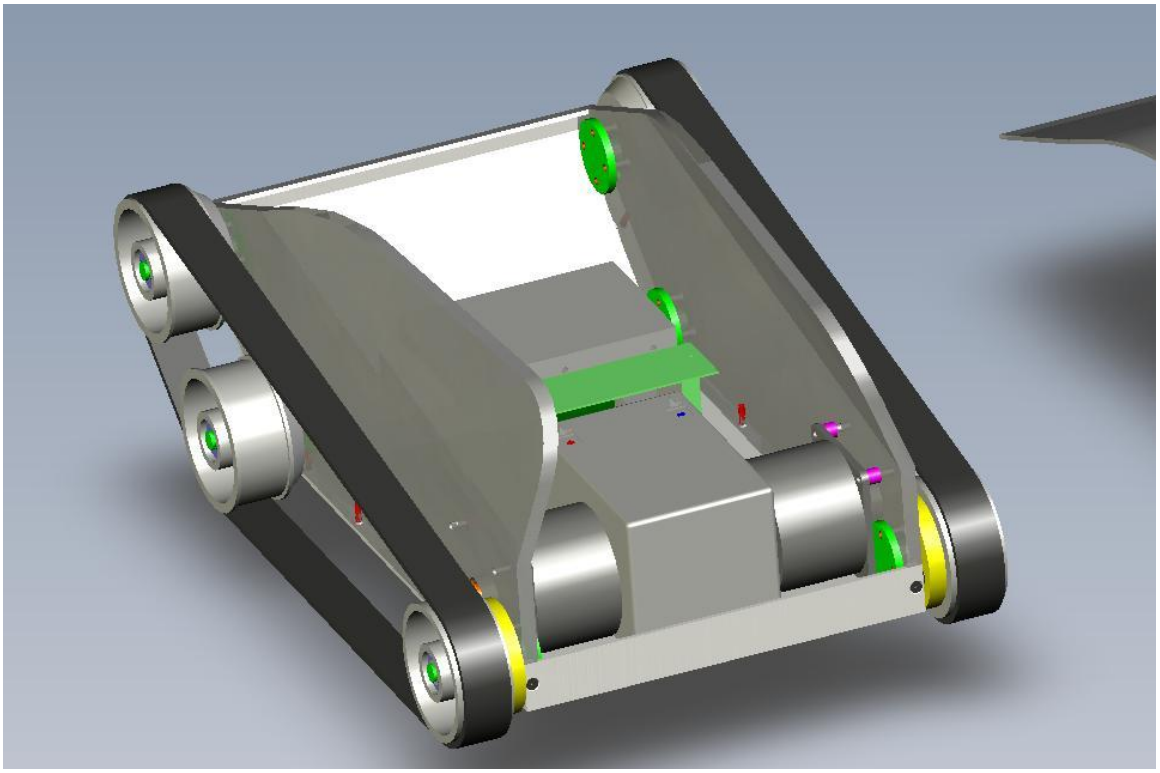
Şekil 3.2 Mobil keşif robotunun gerçek görüntüsü

Robotun manevra kabiliyetinin arttırılması ve engebeli arazilerde de hareket edebilmesinin sağlanması amacıyla, günümüzde kullanımı gittikçe yaygınlaşmakta olan bomba imha robotlarının mekanik yapıları da göz önünde bulundurularak, Şekil 3.2'deki tank benzeri paletli bir araç tasarımı tercih edilmiş ve araç kontrolü bu mekanik yapıya uygun olacak şekilde tasarlanmıştır. Bu yapı geliştirilmeye açık olup, yapılacak mekanik ve elektronik değişiklikler ile birçok farklı alanda kullanılmaya hazır hale getirilebilir. Buna örnek olarak bomba imha robotları, yangın söndürme robotları, depo otomasyonlarında kullanılabilecek taşıyıcı robotlar, askeri amaçlı gözlem ve taarruz robotları gösterilebilir.

Robotun kontrolü yalnızca manuel olarak değil aynı zamanda sanal bir harita üzerinde çizilen bir yol üzerinde otomatik olarak gidecek şekilde de yapılabilmektedir. Bu sayede koordinatları hakkında önceden bilgi sahibi olunan alanlardaki hareketler otomatize edilebileceği gibi, bilgi sahibi olmadığımız alanlardaki hareketler manuel olarak yapılabilmektedir.

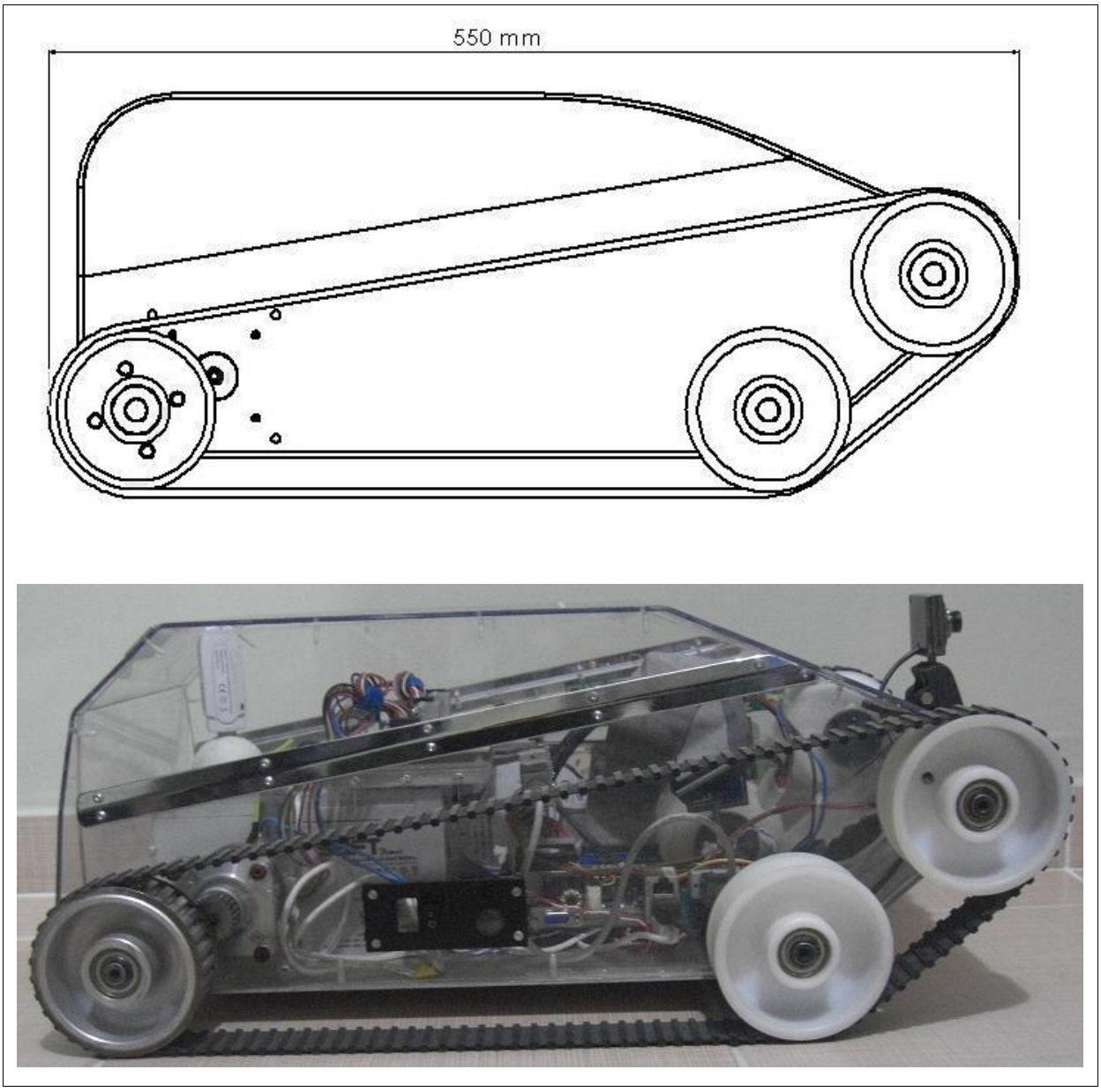
Robot üzerinde konumlandırılan ultrasonik mesafe algılayıcısı yardımıyla robotun içinde bulunduğu alan hakkında ek bilgilere sahip olması sağlanmıştır. Bu projenin konusuna dahil

olmamakla birlikte, yazılım da yapılacak değişiklikler ve geliştirilecek algoritmalar sayesinde içinde bulunduğu ortamın krokisini çıkartarak kullanıcıya iletebilecek bir yapı tasarlamak da mümkündür. Şekil 3.3'te mobil keşif robotunun üç boyutlu iç görünümü verilmektedir.

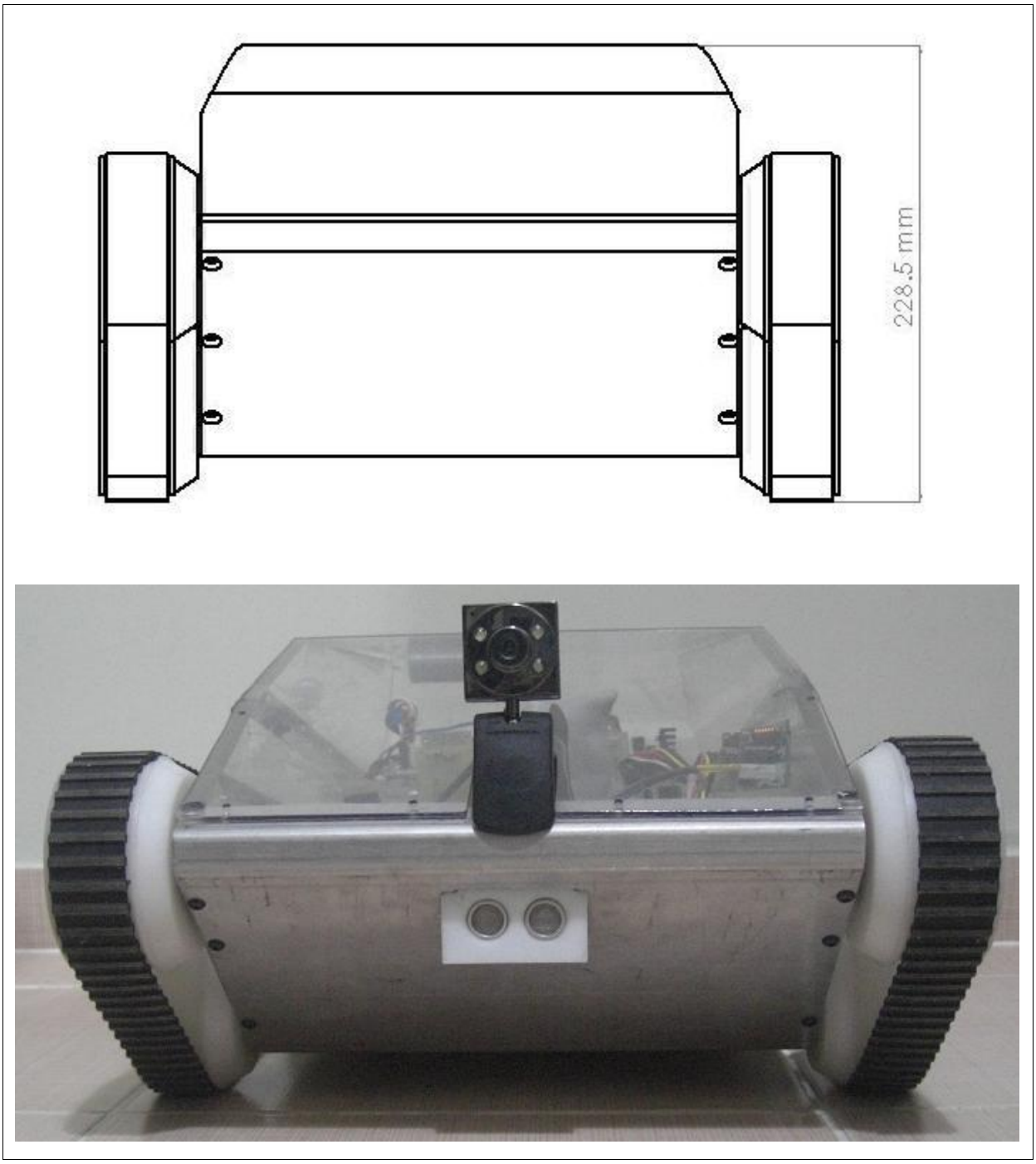


Şekil 3.3 Mobil keşif robotunun üç boyutlu iç görünümü

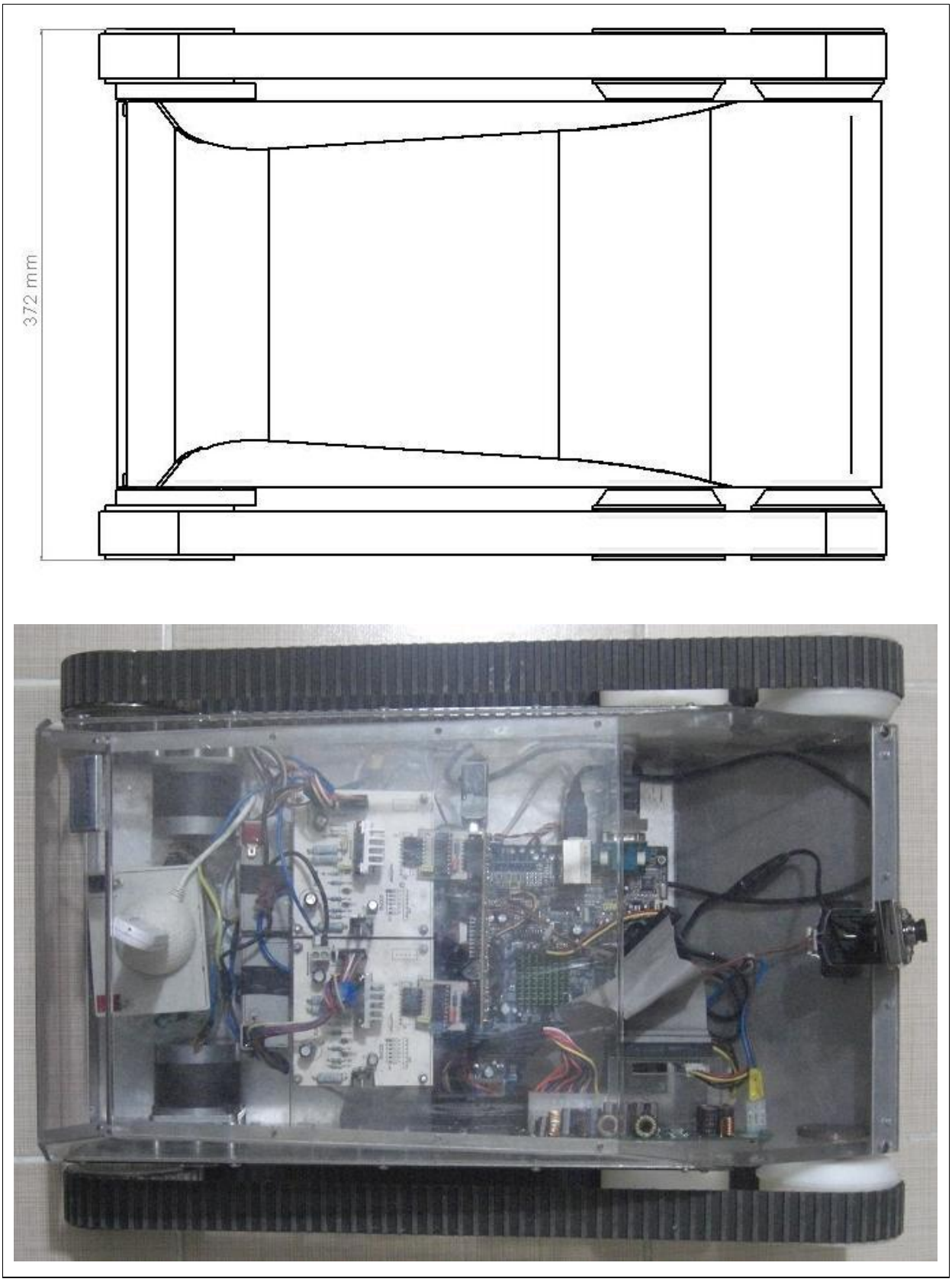
Yandan görünümü Şekil 3.4'te, önden görünümü Şekil 3.5'te ve üstten görünümü Şekil 3.6'da verilmekte olan mobil keşif robotu 550 mm uzunluğa, 372 mm genişliğe ve 228 mm yüksekliğe sahiptir.



Şekil 3.4 Mobil keşif robotunun yandan görünümü



Şekil 3.5 Mobil keşif robotunun önden görünümü



Şekil 3.6 Mobil keşif robotunun üstten görünümü

Mobil keşif robotu maksimum hızda hareket ederken dakikada 12 metre yol alabilmektedir.

Buna göre mobil robotun maksimum hızı 720 m/sn olarak ölçülmüştür.

3.1 Robotun Tasarım Aşamaları

Mobil robot tasarımının ilk aşamasında robotun sahip olması istenilen özelliklerin neler olduğu belirlenmiş ve bu özelliklerin gerçekleştirilebilmesi için kullanılacak malzeme ve yöntemler araştırılmıştır. Proje başlangıcında öngörülen bazı yöntemler, projenin ilerleyen safhalarında karşılaşılan donanımsal ve yazılımsal birtakım problemler nedeniyle değiştirilmek durumunda kalınmış ve çözüme yönelik farklı yöntemler geliştirilmiştir.

Yazılım tasarımına başlanırken karar verilmesi gereken ilk adım, mikrodenetleyici ve bilgisayarlar üzerinde çalışacak kontrol programlarının hangi programlama dili kullanılarak yazılacağının belirlenmesi olmuştur.

Elektronik kartlar üzerinde kullanılan Atmel firmasına ait AVR serisi mikrodenetleyicilerin programlanması Assembly ya da C programlama dilleri kullanılarak yapılabilmektedir. Assembly ile kullanılarak yazılan programlar makine diline daha yakın olduklarından daha performanslı çalışmalarına rağmen büyük boyutlu ve karmaşık programların yazılmasında oldukça zahmetli olmaktadırlar. Bu tip programlarda, daha kabiliyetli ve hazır kütüphanelere sahip olan bir programlama dilinin kullanılması daha uygun olduğundan projede C programlama dili tercih edilmiştir.

Elektronik devreler üzerindeki mikrodenetleyicilerden birinin ultrasonik mesafe algılayıcısının kontrolünü ve algılayıcıdan gelen işaretlerin işlenerek, mesafe ölçümünü yapması sağlanmıştır. Yapılan bu ölçümün sonucu RS232C protokolü vasıtasıyla robot bilgisayarına aktarılmıştır. RS232C protokolü ile ilgili ayrıntılı bilgi Ek-1'de verilmiştir.

Diğer mikrodenetleyici üzerinde yazılan program vasıtasıyla RS232C protokolü ile robot bilgisayarından alınan hareket bilgileri motor adım sayısı ve yön bilgisi olarak ayrıştırılmış ve buna uygun olarak adım motorların sürülmesi sağlanmıştır. Ayrıca yine aynı mikrodenetleyici üzerindeki programda yapılan geliştirmelerle robot önündeki engellere olan mesafenin belirli bir seviyenin altına inmesi durumunda motor hareketinin bilgisayardan gelen hareket emirlerinden bağımsız olarak duraklatılması ve ancak uygun şartlar sağlandığı takdirde hareketine devam etmesi sağlanmıştır.

Robot ve kullanıcı bilgisayarları üzerinde çalıştırılacak olan yazılımlar kullanıcı arayüzüne sahip olacaklarından görsel bir programlama dili ile yazılmaları gerekmektedir. Bu amaca yönelik olarak kullanılabilecek Visual Basic, Delphi, Java ve .NET gibi birbirlerinden farklı

avantaj ve dezavantajlara sahip birçok programlama dili mevcuttur. Bunlar arasında seçim yapılırken çalışabilecekleri platform çeşitliliği, programlama dillerinin lisans durumları, yapılacak projeye uygun hazır kütüphanelerinin kapsamı gibi birçok faktör bulunmaktadır. Bu projede gerek ücretsiz derleyici ve editörlere sahip olması, gerek birçok örnek programın kolayca bulunabilmesi ve gerekse robot bilgisayar üzerinde çalıştırılabilecek farklı işletim sistemleri ile uyumlu olması nedeniyle Java programlama dili tercih edilmiştir.

Yazılım tasarımının ilk aşamasında kullanıcı ile robot programlarının haberleşmesi için kullanılacak sunucu-istemci mimarisi ile ilgili çalışmalar yapılmıştır. Bu mimari ile çalışan örnek sohbet programları incelenmiş ve bu mantıkla temel bir haberleşme programı yazılmıştır. Bu programlar her iki taraftaki uygulamaların birbirlerine metin bazlı mesajlar göndermesine esasına dayanmaktadır.

Sunucu-istemci programlarının haberleşmesi için kullanılacak yöntemin belirlenmesi ihtiyacı bu noktada ortaya çıkmıştır. Günümüzde oldukça yaygın olarak kullanılan Wi-Fi haberleşmesinin projenin amaçlarına en uygun haberleşme yöntemi olduğuna karar verilmiştir. Wi-Fi, IEEE 802.11 standardına dayanan kablosuz bir haberleşme teknolojisidir. Wi-Fi teknolojisi hakkında ayrıntılı bilgi Ek-2'de verilmiştir.

Mobil robot ile kullanıcı arasındaki Wi-Fi haberleşmenin sağlanması için kullanıcı tarafındaki kontrol programı bir bilgisayar üzerinde çalıştırıldığı gibi, robotun bulunduğu sistemin de bir bilgisayar olarak tasarlanmasına ve kullanıcı ile robot programlarının sunucu-istemci mimarisiyle çalıştırılmasına karar verilmiştir.

İkinci aşamada, her iki taraftaki uygulamanın birbirlerinin çalışma durumundan haberdar olması için bir yapı geliştirilmiş ve uygulamalardan birinin herhangi bir nedenle sonlanması durumunda diğer uygulamanın yeni bir bağlantı kurulması için gerekli hazırlıkları yapması ve diğer uygulamanın yeniden çalışmasını beklemeye başlaması sağlanmıştır.

Daha sonraki aşamada, kullanıcı kontrol programının sahip olacağı arayüzün tasarımı gerçekleştirilmiştir. İlk olarak, iki farklı kullanım modu ortaya çıkartılmış ve bunlardan harita modu için uygun arayüzün nasıl olması gerektiğine karar verilmiştir. Buna göre, ekran üzerinde ızgaralara ayrılmış, 4 metre eninde, 6 metre boyunda sanal bir harita oluşturulmuş ve bu harita üzerinde yapılan işaretlemelerin hesaplamaları yapılarak ilgili sonuçlar robot programına gönderilmiştir. Daha sonra klavye modu olarak isimlendirilen ikinci mod tasarımına geçilmiş ve klavyedeki yön tuşları yardımıyla belirtilen yön bilgisi yine metin bazlı olarak robot programına gönderilmiştir.

Robot programına gönderilen kontrol bilgilerinin alınıp işlenmesi amacıyla robot bilgisayarı üzerinde çalışan uygulamada birtakım geliştirmeler yapılmıştır. Buna göre, kontrol programından gelen hareket bilgilerine göre motorların atması gereken adım sayıları hesaplanmış ve hesaplama sonuçları uygun bir biçimde bilgisayarın seri portu aracılığıyla elektronik kontrol kartlarına gönderilmiştir.

Bir sonraki aşamada, robot bilgisayarına bağlanan kamera ve mikrofon yardımıyla ortamdaki görüntü ve ses bilgileri alınmış ve robot kontrol programı tarafından oynatılmıştır. Bunun için java programlama dilinin sahip olduğu JMF görüntü ve ses işleme kütüphaneleri kullanılmış, işlenen bu bilgiler yine bu kütüphanelerde bulunan iletişim sınıfları kullanılarak RTP protokolü ile kullanıcı programına aktarılmış ve kullanıcı programı tarafından oynatılması sağlanmıştır.

3.2 Robot İşletim Sistemi

Tasarımı gerçekleştirilen robot projesinde, robot üzerindeki çevre birimlerin sürücülerine erişim ve bu sürücülerin kurulum kolaylığı göz önüne alınarak kişisel bilgisayarlarda yaygın olarak kullanılan Windows XP işletim sistemi tercih edilmiştir. Ancak bu kısımda alternatif işletim sistemleri hakkında bilgi verilmiş ve bunlardan Linux işletim sisteminin Windows işletim sistemi ile detaylı karşılaştırılması verilmiştir.

3.2.1 Gömülü Linux

Gömülü Linux, linux işletim sisteminin cep telefonları, PDA'ler, elde taşınabilir medya oynatıcıları ve diğer tüketici elektroniği cihazları gibi gömülü sistemlerde kullanılan adıdır.

Geçmişte gömülü sistemler için yapılan yazılımlar doğrudan assembler dili yazılarak geliştiriliyordu. Geliştiriciler tüm donanım sürücülerini ve arayüzlerini baştan geliştirmek zorundaydı. Daha sonraki uygulamalarda, küçük bir ücretsiz yazılım seti ile desteklenmiş Linux çekirdeği gömülü cihazların, sınırlı donanım alanlarına sığdırılabildiği çalışmalar gerçekleştirilmişti. Tipik bir gömülü Linux dağıtımı 2 MB alan üzerinde yer kaplamaktadır. Gömülü Linux'un diğer gömülü işletim sistemlerine olan avantajları açık kaynak kodlu olması, küçük yükleme alanı (2 MB civarı), genellikle ücretsiz olması ve kurulum başına lisans bedeli talep etmemesi, olgun ve kararlı olması (10 yıldan fazla bir süredir pek çok cihazın içinde kullanılmakta) ve iyi desteklenmesi olarak özetlenebilir.

3.2.2 Windows CE

Windows CE, Microsoft tarafından taşınabilir cihazlar için yazılmış bir işletim sistemidir. Windows CE, sanılanın aksine küçültülmüş bir Windows değil özel olarak yazılmış, ayrı bir işletim sistemidir. Windows'un küçültülmüş sürümlerine bir örnek olarak Windows XP Embedded sayılabilir.

Windows'un küçültülmüş bir sürümü olmadığı için normal Windows programları Windows CE altında kullanılamazlar. Bunun diğer bir sebebi de, Windows CE işletim sisteminin çok farklı işlemci mimarilerinde çalışabilir olmasıdır. Öte yandan, daha sonra da açıklanacağı üzere Windows CE için program yazması Windows için program yazmaya fazlasıyla benzediğinden, bazı yazılımların Windows CE sürümü de vardır.

Windows CE, birçok alanda kullanılabilir:

• Ucuz taşınabilir PC'ler

• Çok küçük PC'ler

• Pocket PC'ler

• Cep telefonları

• TV setleri

• Benzer elektronik araçlar

Bir programcı için Windows CE'nin en büyük avantajı, önceden tanıdığı Win32 arayüzüne çok benzer bir arayüz sunmasıdır. Diyalog pencereleri, registry ve DirectX Windows CE'de aynı ya da oldukça ufak değişikliklerle kullanılabilir. Windows CE'de asenkron işlemler bulunmamaktadır. Windows CE üzerinde .NET Compact Framework ile de uygulama geliştirilebilir. Windows altında program yazmak için sıkça kullanılan Visual Studio aracı, ya da gömülü Visual C++, Windows CE'ye de uygun kod üretebilir ve derleyebilir. Bunun için Microsoft'tan gerekli yazılım geliştirme kiti indirilebilir. Gömülü Visual C++ ve yazılım geliştirme kitleri ücretsizdir. Windows CE emülatörü ile geliştirilen yazılım alete aktarılmadan önce test edilebilir.

Windows CE, aşağıdaki yazılım ve donanımları desteklemektedir:

• Word, Excel, Outlook ve son olarak PowerPoint'in Windows CE sürümleri vardır

• Internet Explorer Mobile başta olmak üzere bazı web tarayıcılarını kullanılabilir.

• MSN Messenger'ın Windows CE sürümü vardır

• Opera Web Browser, 2006 yılında 4 farklı işlemci mimarisinde çalışmak üzere Windows CE için yeni tarayıcı çekirdeğini kullanan iki ayrı web tarayıcı ve web tarayıcı yazılım geliştirme kiti hazırlamıştır

• Windows Media Player'a ek olarak birçok alternatif yazılım sayesinde birçok çoklu ortam dosyası açılabilir

• Windows CE, aygıt sürücüsü bulunduğu takdirde Wi-fi, Bluetooth, GSM, GPRS, EDGE, 3G veya VPN gibi sayısız bağlantı desteğine sahiptir

• Birçok çevre birim (USB diskler gibi) desteklenir

Windows CE, gerçek zamanlı işletim sistemidir. Pocket PC 2002, Pocket PC 2003 ve Windows Mobile 5.0 işletim sistemlerinin temeli Windows CE'ye dayanır. Windows CE, farklı işlemci mimarileri üzerinde, 1MB gibi oldukça küçük hafizayla çalışabilir.

3.2.3 Windows XP Embeded

Windows XP Embedded ya da diğer adıyla XPe, Microsoft tarafından gömülü sistemler için tasarlanmış bir işletim sistemidir. Mevcut Windows uygulamalarını ve cihaz sürücülerini çalıştırabilmektedir.

XPe' nin Windows CE ile herhangi bir bağlantısı yoktur. İkisi de farklı cihazları hedeflemekte ve her birinin OEM üreticilerinin dikkate aldığı eksi ve artı yönleri bulunmaktadır. Örneğin, XPe hiçbir zaman CE' nin üzerinde çalıştığı kadar az yer kaplamayacaktır. Bununla birlikte, CE XPe' nin kullandığı Win32 API' lerini kullanamayacak ve mevcut sürücü ve uygulamaların çok büyük kısmını çalıştıramayacaktır.

XPe' nin kullanımının hedeflendiği cihazlar, ATM' ler, yiyecek ve meşrubat otomatları, yazar kasalar, atari makineleri, endüstriyel robotlar, ince istemciler, elektronik ev eşyaları olarak gösterilebilir.

İşletim sisteminin kullanıcı tanımlı versiyonları PC dışında her yerde kullanılabilir. XPe, XP Pro ile aynı donanımı (x86 mimarisi) destekleyebilmesine rağmen lisans sınırlamalarından dolayı standart PC'lerde kullanılmamaktadır.

3.2.4 Windows ve Linux İşletim Sistemlerinin Karşılaştırılması

Bu bölümde Windows ve Linux işletim sistemlerinin, toplam maliyetleri, pazar payları,

kullanıcı arayüzleri, kurulumları, erişebilirlik ve kullanılabilirlikleri, kararlılık ve destek durumları karşılaştırmalı olarak verilmiştir.

Çizelge 3.1 Windows ve Linux işletim sistemlerinin karşılaştırılması

	WINDOWS	**LINUX**
Toplam Maliyet		
İlk alım	Sürüme göre 45 dolardan 450 dolara.	Sürüm ve dağıtıma göre 0 dolardan 350 dolara.
Destek	199 dolardan 7.999 dolara yıllık.	0 dolardan 3.702 dolara
Anti virüs yazılımı	Yıllık bilgisayar başına 0 dolardan 100 dolara.	Gerekmiyor.
Pazar payı		
Yaklaşık masaüstü kullanım payı	Mayıs 2009 itibariyle %87.90.	Mayıs 2009 itibariyle, %1.02.
Ön yüklü olma	Neredeyse tüm kişisel bilgisayarlarda ön yüklü.	Birkaç kişisel bilgisayarda ön yüklü. Ubuntu tüm System76 bilgisayarlarda, bazı Dell bilgisayarlarda yüklü.
Kullanıcı arayüzü		
Grafiksel kullanıcı arayüzü	Windows Kabuğu, Desktop Windows Manager'i Vista'da pencere yöneticisi olarak kullanır.	Çeşitli masaüstü ortamları mevcut ve en çok kullanılanları GNOME ve KDE. KDE dördüncü sürümüyle birlikte masaüstü ortamına yeni bir soluk getirdi ayrıca Linux'un üç boyutlu masaüstü görüntüleri de pek çok kişi tarafından beğenilmektedir.
Komut satırı arayüzü	Komut satırı arayüzü kullanıcı ve işletim dizgesi arasında doğrudan iletişim sağlamak için vardır.	Linux konsolla sıkıca bütünleşmiştir ve çok iyi bir konsol ortamı sunar. Grafik dizge çökse bile konsolla bundan kurtulmak mümkündür. Çeşitli Unix kabukları mevcuttur.
Kurulum		
Kurulum kolaylığı	Windows Server 2003 ve önceki sürümler iki bölüme ayrılır, metin tabanlı kip ve ardından grafiksel kip.	Dağıtıma göre değişir fakat pek çok dağıtım yeni ve deneyimsiz kullanıcılar için basit grafik kullanım araçları sunar.
Kurulum süresi	Sürüme ve donanıma göre değişir fakat genelde 20 dakikadan bir saate kadar sürer.	Dağıtma bağlıdır. Paket tabanlı kurulumlar (Ubuntu, Debian) kısa sürer.

Aygıt sürücüsü	Windows genelde işletim dizgesini işlevsel hale getirecek kadar sürücü içerir.	Linux çekirdekleri pek çok dağıtımda sürücülerin çoğunu yüklenebilir çekirdek modülü olarak içerir.
Çalışan ortamlarla kurulum	Windows Preinstallation Environment veya BartPE ile kurulabilir. Fakat sadece Microsoft sertifikalı dizge kurucularının WinPE diski ile kurulum yapmasına izin verilmiştir. Son kullanıcıların WinPE kullanmasına izin verilmemiştir.	Neredeyse tüm Linux dağıtımlarının bir çalışan cd'si vardır. Herhangi bir lisans kısıtlaması yoktur.
Önyüklü yazılımlar	Bazı çoklu ortam ve yardımcı yazılımlar (Internet Explorer, Not Defteri, Windows Media Player gibi) yüklü gelir. Ancak bir ofis yazılımı veya gelişmiş ortam oynatıcı yoktur.	Tüm büyük dağıtımlar pek çok yazılımla (internet tarayıcı, ofis yazılımı, resim işleme yazılımı vb.) beraber gelir. Bazı dağıtımlar belirli alanlarda özel yazılımlarla birlikte gelir.
Ön yüklü olmayan yazılımlar	Çok büyük bir ticari yazılım havuzu vardır ve programlar ayrı ayrı kurulabilir.	Büyük bir özgür yazılım havuzu ve biraz da ticari yazılım havuzu vardır. Bir Microsoft çalışanı 1998 yılında yayınlanan bir raporda "İnsanların Linux'a geçerken gereksinim duyduğu pek çok ticari yazılımın zaten özgür sürümleri mevcut" diye belirtmiştir.
Bölümlendirme	NTFS bölümleri sorun yaratmadan genişletmek mümkün. Bölümlendirme için üçüncü parti gelişmiş yazılımlar da mevcut.	Bazı dosya dizgeleri veri kaybetmeden yeniden boyutlandırmayı destekliyor. Tüm Linux dağıtımlarında fdisk veya gparted gibi disk bölümlendirme yazılımları var.
Önyükleyici	Birden fazla işletim dizgesini yükleyebilir.	Birden fazla işletim dizgesini yükleyebilir.
Erişilebilirlik ve kullanılabilirlik		
Kullanıcı odaklılık	Genelde tutarlıdır.	Grafik tasarımın niteliği dağıtımlarda ve masaüstü ortamlarında değişiklik gösterir.
Sürümler arası tutarlılık	Sürümler arasında kullanıcının yazılımla etkileşimi genelde tutarlıdır.	Genelde tutarlıdır fakat yazılımlar kullanıcı tarafından büyük oranda değiştirilebilir.

Kararlılık		
Genel kararlılık	NT tabanlı Windows sürümleri teknik olarak eski sürümlerden (95, 98, ME) daha kararlıdır.	Çekirdek teknik olarak Unix'in kararlılığını modüler mimariden dolayı miras almıştır. Pencere yöneticilerinin kararlılığı değişkendir fakat genelde kararlıdırlar.
Aygıt sürücüsü kararlılığı	Aygıt sürücüleri ya Microsoft tarafından sağlanır ya da donanım üreticileri sağlar.	Aygıt sürücülerinin Linux'ta çalışması tersine mühendislik yöntemiyle sağlanır. Intel ve HP gibi bazı üreticiler özgür sürücüler üretmektedirler.
Açılış ve kapanış	Sürücü güncellemesi ve dizge güncellemesi sonrası yeniden başlatmak gerekir.	Sadece çekirdek sürücü modülleri değiştirilecekse yeniden başlatmak gerekir.
Geri yükleme	Programlar CTRL+SHIFT+ESC veya CTRL+ALT+DEL ile sonlandırılabilir.	KDE uygulamaları CTRL+ALT+ESC veya KSystemGuard ile kapatılır.
Donanım soyutlama katmanı	Windows NT4, 2000 ve sonrası sürümlerde mevcut. Windows 95, 98'de bulunmuyor.	Pek çok Linux dağıtımında mevcut.
Destek		
Topluluk desteği	MSDN gibi topluluk siteleri var.	Destek genelde ileri kullanıcılar ve geliştiriciler tarafından internette sağlanıyor.
Telefon desteği	Microsoft veya OEM tarafından.	Red Hat, Canonical, Novell gibi büyük dağıtımcılar telefon desteği sağlamaktadır.
Belgelendirme	Geniş bir belgelendirme internette ve kitaplarda mevcuttur.	Belgeler genelde internette sıkça sorulan sorular veya viki sayfalarında mevcuttur.
Alıştırma	Pek çok bilişim kursunda Windows öğretilmektedir.	Linux genelde üniversitelerde bilgisayar derslerinde öğretilmektedir. Bazı Linux sertifikaları da büyük dağıtımcılar tarafından verilmektedir.
Üçüncü parti belgelendirme	Windows'un pazar payı fazla olduğundan pek çok üretici programlarının Windows'a özel kurulum ve kullanım belgelerini sağlamaktadır.	Linux için olmayan programlar Linux ile ilgili bir belgelendirme sağlamamaktadır.

3.3 Robot Maliyeti

Projenin geliştirme safhasında yapılan denemeler sırasında kullanılan malzemeler için yapılan harcamalar göz ardı edilerek çıkarılmış yaklaşık maliyet listesi Çizelge 3.2'de verilmiştir.

Çizelge 3.2 Robot yaklaşık maliyet çizelgesi

Malzeme	**Fiyat**
Şase için alüminyum ve pleksi malzeme	300 TL
2 adet alüminyum kasnak	50 TL
2 adet palet	70 TL
1 adet anakart	200 TL
3 adet 12V 4A akü	60 TL
1 adet 2 GB CF kart	20 TL
1 adet CF-IDE dönüştürücü	20 TL
1 adet USB kablosuz ağ adaptörü	70 TL
1 adet USB kamera	20 TL
1 adet DC-DC çevirici	60 TL
1 adet SRF05 ultrasonik algılayıcı modül	50 TL
2 adet adım motor	50 TL
2 adet adım motor sürücü kartı	45 TL
1 adet elektronik kontrol kartı	20 TL
2 adet 128MB SD Ram	50 TL
TOPLAM	1.085 TL

Bu maliyetler araştırma ve geliştirme faaliyetlerini de içermektedir ve ürünlerin perakende fiyatları baz alınarak hesaplanmıştır. Seri üretime geçilmesi durumunda malzemelerin fiyatları daha uygun olacak ve dolayısıyla maliyetin de daha düşük olması beklenmektedir.

4. MOBİL KEŞİF ROBOTUNUN DONANIMI

Mobil keşif robotunun donanımı Yiğiter'in (2010) "Mobil Keşif Robotu Tasarımı" isimli yüksek lisans tez çalışmasında detaylı olarak anlatılmış olmakla beraber bu kısımda donanım ile ilgili özet bilgilere yer verilmiştir.

4.1 Donanımın Blok Yapısı

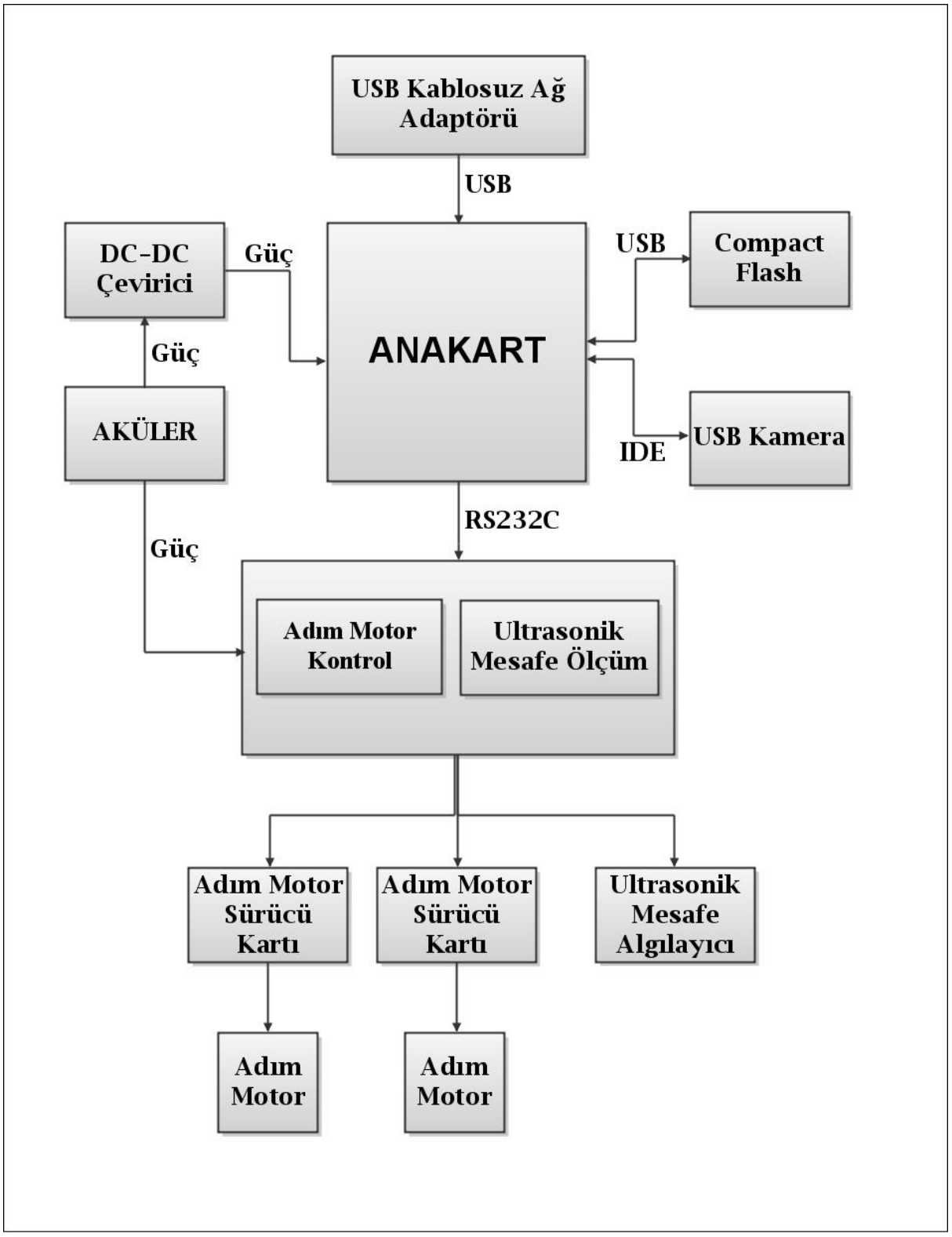
Mobil keşif robotunun elektronik donanımı Şekil 4.1'de de gösterildiği gibi farklı işlevleri gerçekleştiren bloklara sahip bir sistem şeklinde tasarlanmıştır. Bu tasarım şekli ileride yapılması muhtemel geliştirmeler için uygun bir zemin hazırlamaktadır. Örneğin, yapılması planlanan bir projede adım motor yerine DC motor kullanılması ihtiyacı doğacak olursa, sistemin tamamını değiştirmek yerine ilgili bloklar değiştirilerek ve yazılımda ufak düzenlemeler yapılarak istenilen yapıya geçişin sağlanması mümkündür.

Sistemdeki blokların her birisi farklı işlevsel özellikleri yerine getirmekte ve gerekli durumlarda bu bloklar birbirleri ile veri alışverişi yapmaktadır.

Bu projede kullanılan elektronik donanımın malzeme listesi Çizelge 4.1'de verilmektedir.

Çizelge 4.1 Elektronik donanım malzeme listesi

Parça	**Adet**
VIA Epia V Serisi Mini-ITX anakart	1
Airties WUS-300 USB Wi-Fi adaptör	1
Compact Flash bellek kartı	1
Fly USB web kamera	1
Mini-ITX DC-DC çevirici	1
IBM adım motorları	2
12V 4A akü	3
SRF05 Ultrasonik mesafe algılayıcı modülü	1
Elektronik kontrol kartları	1

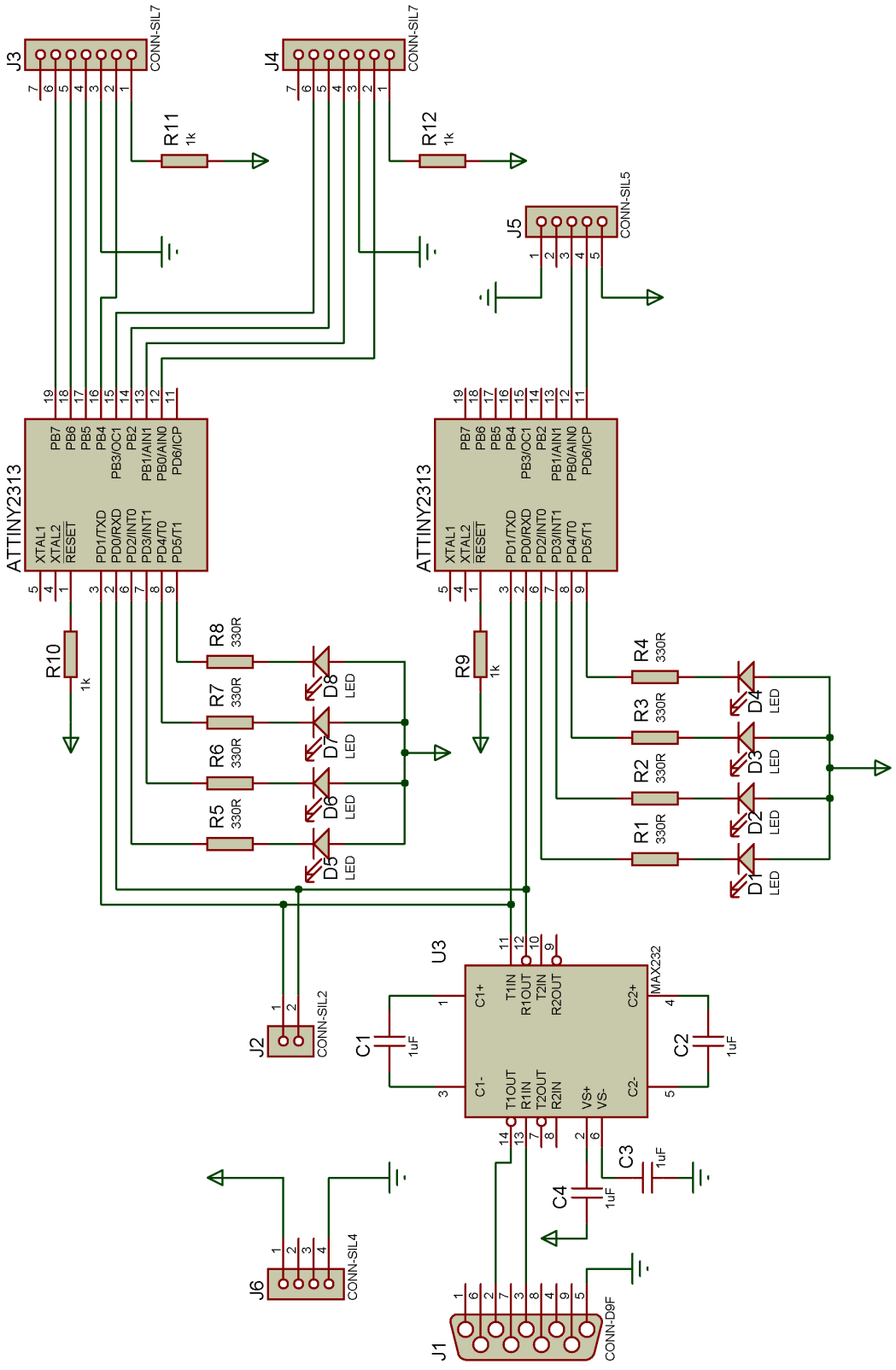


Şekil 4.1 Elektronik donanım blok şeması

4.2 Elektronik Kontrol Kartı

Bilgisayardan gelen hareket bilgilerine uygun olarak motorların sürülmesini sağlayan ve

ultrasonik mesafe ölçüm algılayıcısından elde ettiği mesafe bilgisini robot bilgisayarına aktaran elektronik kontrol kartının devre şeması Şekil 4.2'de verilmiştir.



Şekil 4.2 Kontrol kartı devre şeması

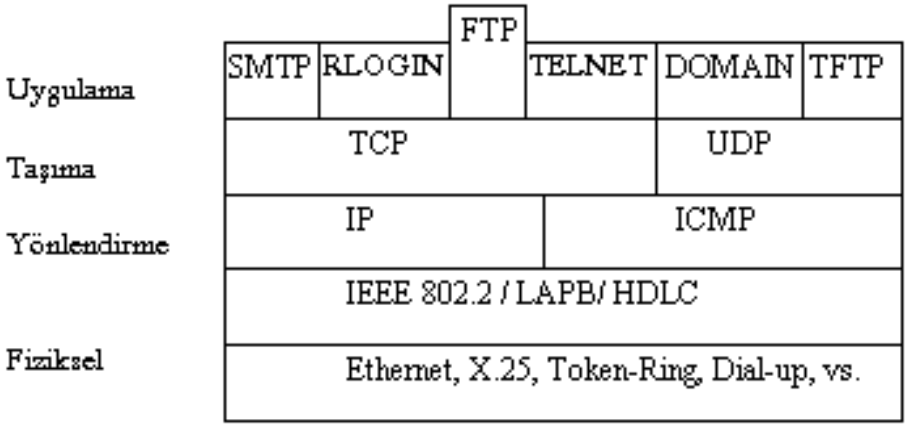
5. MOBİL KEŞİF ROBOTUNUN YAZILIMI

Mobil keşif robotu, yazılım aracılığıyla kontrol edilen elektronik bir donanım olacak şekilde tasarlanmıştır. Projenin önemli bir bölümünü de yazılım oluşturmaktadır. Bu bölümde robot bilgisayar üzerinde çalışan java tabanlı kontrol programı ve kullanıcının robotu kontrol etmesi için bir arayüz olarak tasarlanan kullanıcı kontrol programı ve elektronik kontrol kartları üzerinde çalışan programlar hakkında detaylı bilgi verilmiştir.

5.1 TCP/IP ve Sunucu-İstemci Mimarisi

Tasarlanan mobil keşif robotunda robot ve kullanıcı kontrol programlarının haberleşmesi için TCP/IP protokolü kullanılmıştır.

TCP/IP katman modeli aynı zamanda internet katman modeli veya internet başvuru modeli olarak da bilinir. TCP/IP protokol yığını kullanarak çalışmaktadır. Bu yığın iki makine arasında veri aktarımı için gereken bütün protokolleri içerir. TCP/IP protokolünün katmanları Şekil 5.1'de görülmektedir.



Şekil 5.1 TCP/IP katmanları

Java dili ile geliştirilen ağ programları genelde uygulama katmanında çalışır. Aktarım katmanındaki temel TCP ve UDP protokol bilgileri, bağlantılı ve bağlantısız istemci/sunucu ağ yazılımları geliştirirken faydalıdır. Aktarım seviyesindeki protokoller TCP ve UDP olmak üzere iki tanedir. Java dili her iki protokolü de desteklemektedir. Programcı hangisini kullanacağını kendisi tercih etmektedir. TCP/IP iletişiminde bir uç nokta, bir IP adresi ve bir port numarası çifti ile tanımlanan bir soket ile belirlenir.

TCP protokolü connection-oriented olarak adlandırılan ve iki bilgisayar arasında veri transferi

yapılmadan önce bağlantının kurulması ve veri iletiminin garantili olarak yapıldığı bir protokoldür. TCP iletişiminde veri paketleri kullanılır. Ayrıca gönderen ve alan uygulamalarda da port bilgisi eklenir. Port (çıkış), kaynak ve hedef uygulamanın iletişimini sağlar. TCP byte-stream iletişimi kullanır. Bu yöntemde TCP segmentlerindeki veriler bir bayt dizisi olarak işlenir.

TCP internette kullanılan temel protokoldür. Dosya aktarımı ve uzaktan bağlantılar gibi işlem-kritik görevlerde kullanılır. Bazen güvenli stream dağıtımı olarak da belirtilen TCP, verinin gönderildiği sırada ve durumda hedefe varmasını sağlar.

Bir TCP iletişimi kaynak ile hedef makine arasında kurulan bir sanal devre ile gerçekleşir. Bu devre genellikle 3 aşamalı bir işlemle açılır. Devre açıldıktan sonra veri aynı anda her iki yönde de seyahat edebilir. Bu iletişim hattı bazı kaynaklarda full-dublex iletim olarak adlandırılır. Bu iletişimde her iki makinede eş zamanlı haberleşebilir. TCP yoğun hata-kontrol yapıları sağlamaktadır. Her gönderilen veri bloğu için bir nümerik değer üretilir. İki makine her veri aktarımını bu nümerik değere göre belirler. Her başarılı aktarılan bir blok için, hedef makine gönderen makineye aktarımın başarılı olduğuna dair bir mesaj gönderir. Eğer aktarım başarısız ise, iki şey olabilir. İsteyen makine bir hata bilgisi alır veya isteyen makine hiçbir şey almaz. Bir hata alındığında, bu hata iletişimin kesilmesine neden olacak kadar önemli değil ise veri tekrar aktarılır. Hayati bağlantıya tipik bir örnek karşı bağlantının kesilmesidir. Bu durumda paket iletilmeden veri aktarımı durdurulur.

TCP soketlerinin güvenilir ve sıralı yapısından dolayı, bu soketler sık olarak stream soketleri olarak adlandırılır. Bu soketlerde, paketler ve başlıklar düşük düzeyli verilerle ilgilenmeden, veri blokları sıralı sürekli byte dizileri olarak okunabilmekte ve yazılabilmektedir. Java ile TCP soketlerini kullanmak için java.net.ServerSocket ve java.net.Socket sınıflarına gerek vardır.

UDP de bir gönderim katmanı protokoldür. Ancak UDP iletiminde sağlama yapılmadığı için gönderim garantisi olmaz. Broadcast iletiminde, az miktardaki verilerin iletilmesi için UDP paketleri kullanılır. UDP iletimi, gönderimin garanti edilmediği connectionless türü bir iletişim kurar. UDP bağlantısız datagram servisidir. UDP kaybolan verilerin kurtarılması konusunda herhangi bir garanti vermez. Bu nedenle güvenilir bir protokol olarak nitelendirilmez.

UDP alınan verilerin garantisine gereksinim duymayan uygulamalar tarafından kullanılır. NetBIOS name servisleri, NetBIOS datagram servisi ve SNMP servisleri UDP kullanan

uygulamalara örnektir. UDP, IP'ye basit bir uygulama arabirimi olarak hizmet eder. Güvenilirlik, akış kontrol veya hata-bulma ölçümleri olmadığından, prensip olarak, IP'nin alışverişi ve uygulamaların trafiği için bir port multiplexer/demultiplexer gibi hizmet eder.

UDP, datagramları doğru üst katman uygulamalarına yönlendirmek için port kavramını kullanır. UDP datagramı, bir varış port numarası ve bir kaynak port numarası içerir. UDP modülü varış numarasını trafiği doğru alıcıya teslim edebilmek için kullanır. Java ile UDP protokolünü ve UDP soketlerini kullanmak için java.net.DatagramSocket ve java.net.DatagramPacket sınıflarına gerek vardır.

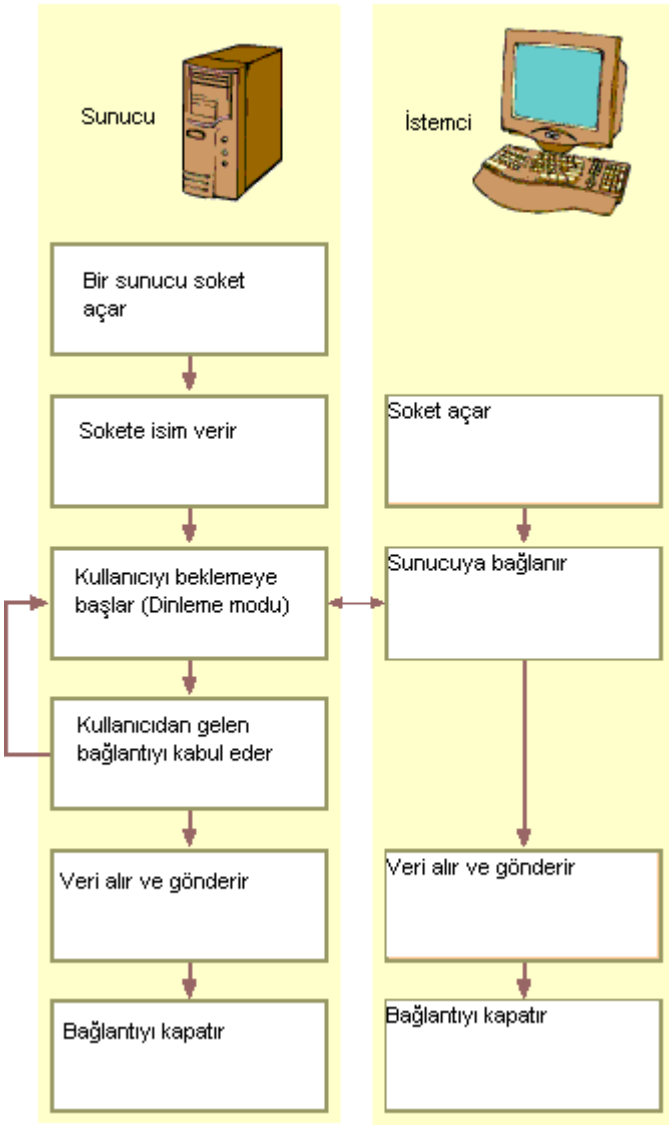
TCP/IP protokolü ile çalışan ağlarda ortak adresleme sağlamak için, TCP/IP protokol yazılımı her makineye bir tek adres atayan bir adresleme yapısı sunar. Kullanıcılar uygulama programları ve protokol yazılımındaki yüksek katmanlar, haberleşme için soyut adresler kullanır.

TCP/IP protokolünde adresleme IP protokolü ile belirlenir. IP standardında her makineye IP adresi olarak belirtilen 32 bit rakam atanır. IP adres hiyerarşisi iki önemli özelliği sağlar:

- Her bilgisayara bir tek adres atanır.
- Ağ numarası atamaları genel olarak ayarlanmasına rağmen, son ekler genel ağ ayarları düşünülmeden atanabilir.

Hedef bilgisayarın network üzerindeki yerini bulur. Paketlerin adreslenmesi ve network üzerindeki bilgisayarlar arasında yönlendirilmesini sağlar. IP iletimi de UDP gibi gönderimin garanti edilmediği connectionless türü bir iletişim kurar. IP, iki bilgisayar (aygıt) paketlerin yönlendirilmesini sağlayan bağlantısız bir protokoldür.

İki bilgisayar arasındaki haberleşmede sunucu-istemci mimarisi esas alınmıştır. Burada robot tarafı bilgisayar sunucu, kullanıcı tarafı bilgisayar ise istemci rolünü üstlenmektedir. Sunucu-istemci haberleşmesinin genel akışı Şekil 5.2'de görülmektedir.



Şekil 5.2 Sunucu istemci haberleşmesi

İstemci ve sunucuların özellikleri aşağıdaki gibi verilebilir.

İstemci:

- Herhangi bir uygulama programı ağ hizmetine gereksinim duyduğunda istemci olur.
- Diğer hesaplamaları da yerine getirir.
- Doğrudan kullanıcı tarafından çağrılır.
- Kullanıcının bilgisayarında yerel olarak koşar.
- Sunucuyla olan bağlantıyı başlatır.

- Aynı anda yalnız birine olmak üzere birden çok hizmete erişebilir.
- Özel bir donanım veya karmaşık işletim sistemi gerektirmez.

Sunucu:

- Ağ hizmeti sağlamaya adanmış özel amaçlı uygulamadır.
- Sistem açılışı zamanında başlar.
- Genellikle merkezi, paylaşılmış uzak bir bilgisayar üzerinde çalışır.
- İstemcilerden hizmet istekleri bekler, sonraki isteği beklemek için döngü yapar.
- Rastgele istemcilerden istek kabul eder, her istemciye bir hizmet sağlar.
- Güçlü donanıma gereksinim duyar.

Bir sunucunun temel yaşam döngüsü ise aşağıda verilen adımları takip eder:

1. Bir kurucu metot yardımı ile belirtilen bir yerel port üzerinden yeni bir ServerSocket oluşturulur.
2. ServerSocket nesnesi accept( ) metodunu kullanarak gelen bağlantı girişimlerini dinler.
3. Bir istemci bağlantı yapmaya girişinceye kadar accept( ) metodu bloklanır. Bağlantı isteği durumunda, accept( ) sunucuya bağlanan bir Socket nesnesi döndürür.
4. Sunucunun istemci ile yapacağı işleme göre, Socket'in getInputStream( ) veya getOutputStream ( ) metodu veya her ikisi de kullanılarak, istemci ile haberleşen giriş ve çıkış streamleri oluşturulur.
5. Sunucu ve istemci aralarında anlaştıkları protokole göre, bağlantıyı kapatma zamanlarına kadar iletişimde bulunur.
6. Sunucu ve istemci veya her ikisi bağlantıyı kapatır.
7. Sunucu adım 2'ye gider ve bir sonraki bağlantı için bekler.

Bir istemcinin temel yaşam döngüsü aşağıda verilen adımları takip eder:

1. Bir kurucu metot yardımı ile belirtilen bir yere port üzerinden yeni bir Socket oluşturulur.

2. Socket nesnesinin oluşturulması ile birlikte istemci uygulaması sunucu uygulamasına bağlanır.

3. Bağlantı kurma işleminden sonra verileri sunucu uygulamasından almak için getInputStream metodunu çağırır. Sunucu ile haberleşen giriş ve çıkış streamleri oluşturulur.

4. Sunucu ve istemci aralarında anlaştıkları protokole göre, bağlantıyı kapatma zamanlarına kadar iletişimde bulunur.

5. Sunucu ve istemci veya her ikisi bağlantıyı kapatır.

5.2 Java Programlama Dili

Cep telefonlarında (PDA vs.) ve diğer hareketli cihazlarda kablosuz internet servislerinin sürekli olarak büyümesi içerik geliştiricilerin ve servis sağlayıcıların yeni kablosuz pazarlar için geliştirilen ürünlere hızlı bir şekilde uyum sağlamalarını gerektirmektedir. Bu gereksinime Java cevap vermektedir ve Java ile yeni aygıtlar çabuk bir şekilde var olan uygulamalar ile desteklenebilir. Geliştiriciler, servis sağlayıcılar ve üreticiler sürekli gelişen kablosuz internet pazarında yer alan yeni uygulamaları kullanmaya anında başlayabilirler. Geçen birçok yıl boyunca, Java internet gelişimi için önemli bir dil olarak ortaya çıkmıştır.

Günümüz kablosuz aygıtları çok çeşitli grafiksel kullanıcı arayüzleri (GUI) ve tarayıcılar kullanmaktadır. Kablosuz ağların potansiyelini tam olarak anlayabilmek için uygulamalar daha bütünleşik hale getirilmelidir. Bu da kablolu ve kablosuz internet arasındaki boşluğu dolduran teknolojilerle çalışmak anlamına gelmektedir. Java da bu teknolojilerden bir tanesidir.

Sun Microsystems tarafından geliştirilmiş bir platform olan Java, bir uygulamanın bir kere yazılmasını ve kablosuz cihazları da içeren değişik platformlarda çalışmasını olanaklı kılar. Avuç içi ve taşınabilir cihazlar için tasarlanmış bir dil olan Java, kendi kavramsal esaslarına geri dönmektedir. Java, internet ve kablosuz ağlar için anahtar bir dildir. Nokia, Ericsson ve LG Electronics hareketli telefonlarında Java'yı desteklemektedir. Amerikan Express, Blue Card teknolojisi için Java'yı kullanmakta ve Sega, Dreamcast oyun konsollarına Java kurmaktadır.

Taşıyıcılar için Java değer katılmış kablosuz veri servisleri için güvenli bir platform sağlamaktadır. Geleneksel girişimci fonksiyonlar ve B2B ticaret uygulamaları kablosuz

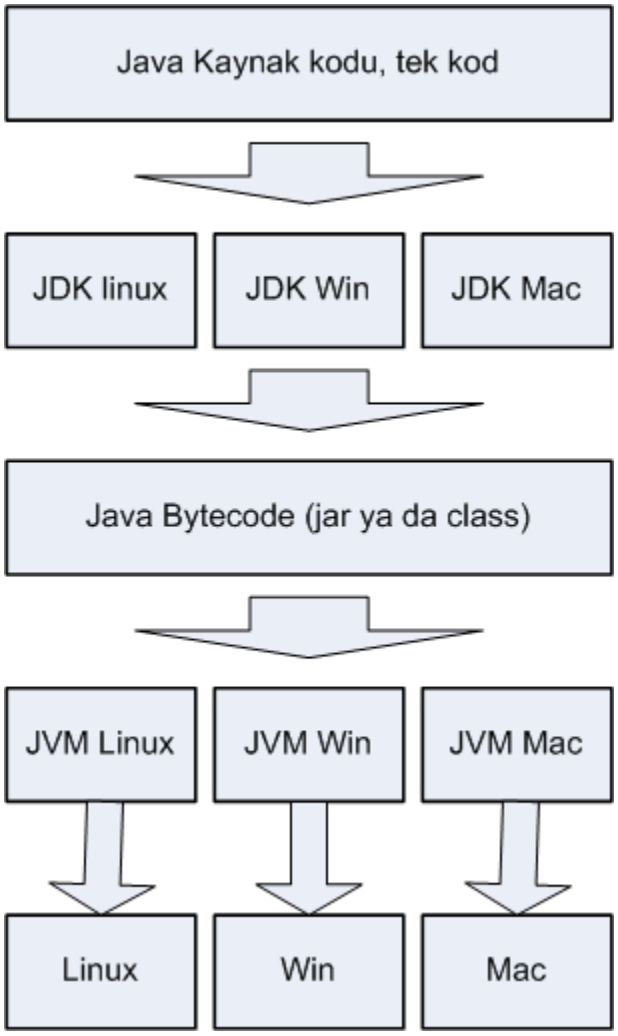
alanlara yayılabilir. Bu sayede yeni bilgi ve eğlence tabanlı uygulamalar avuç içi kablosuz cihazlar tarafından desteklenebilir.

Sadece birkaç sene içinde Java internet uygulamalarının yeğlediği bir dil haline gelmiş ve artık günümüzde de kablosuz hesaplama alanının her dalında kullanılmaya başlanmıştır. Java teknolojisinin bir versiyonu olan J2ME, cep telefonları ve PDA'ler gibi hafıza kısıtlamaları olan aygıtlarda çalışmak üzere tasarlanmıştır. Bir Java motoru ya da Java programları ile ses-merkezli cihazlar, PDA'lerle ve taşınabilir PC'lerle bağlantılı olan bir takım özellikleri yerine getirebilir.

Java'nın kablosuz teknolojide kullanılmasında birçok faktör rol oynamaktadır. Uygulama taşınabilirliği, uygulamaların ve servislerin dinamik dağıtımı, dinamik yükseltmeler (upgrades), geliştirilmiş kullanıcı tecrübesi, bağlantısız erişim ve güvenlik bu faktörler arasında sayılabilir.

Günümüzde Java, sadece bilgisayar ve internet uygulamalarında değil, bilgisayar teknolojilerinin kullanıldığı diğer alanlarda da (akıllı kart, ev teknolojisi ürünleri, beyaz eşyalar vb gibi) programlama ve kontrol aracı olarak kullanılmaktadır. Önümüzdeki dönemlerde, aynı ortak platformu kullanan ama birbirinden farklı gibi görünen cihazların (bilgisayar, elektronik sistemler, elektrikli/elektronik ev eşyaları gibi) bu teknoloji yardımıyla aynı ağ üzerinde bulunacağını ve söz gelimi bilgisayarımızdan evinizdeki birtakım elektronik/elektrikli eşyaları kontrol edebileceğimizi düşünebiliriz.

Java taşınabilir bir dildir. Bunun nedeni de Java programlarının sanal makine (VM-Virtual Machine) olarak adlandırılan bir yazılım tarafından byte kodlarına çevrilmesidir. Java byte kodlarını çalıştıran bu sanal makine PC'lerden UNIX çalışma istasyonlarına, IBM ana bilgisayarlarından cep telefonlarına, PDA'lere ve akıllı kartlara kadar tüm platformlarda kullanılabilir.



Şekil 5.3 Java sanal makinesi

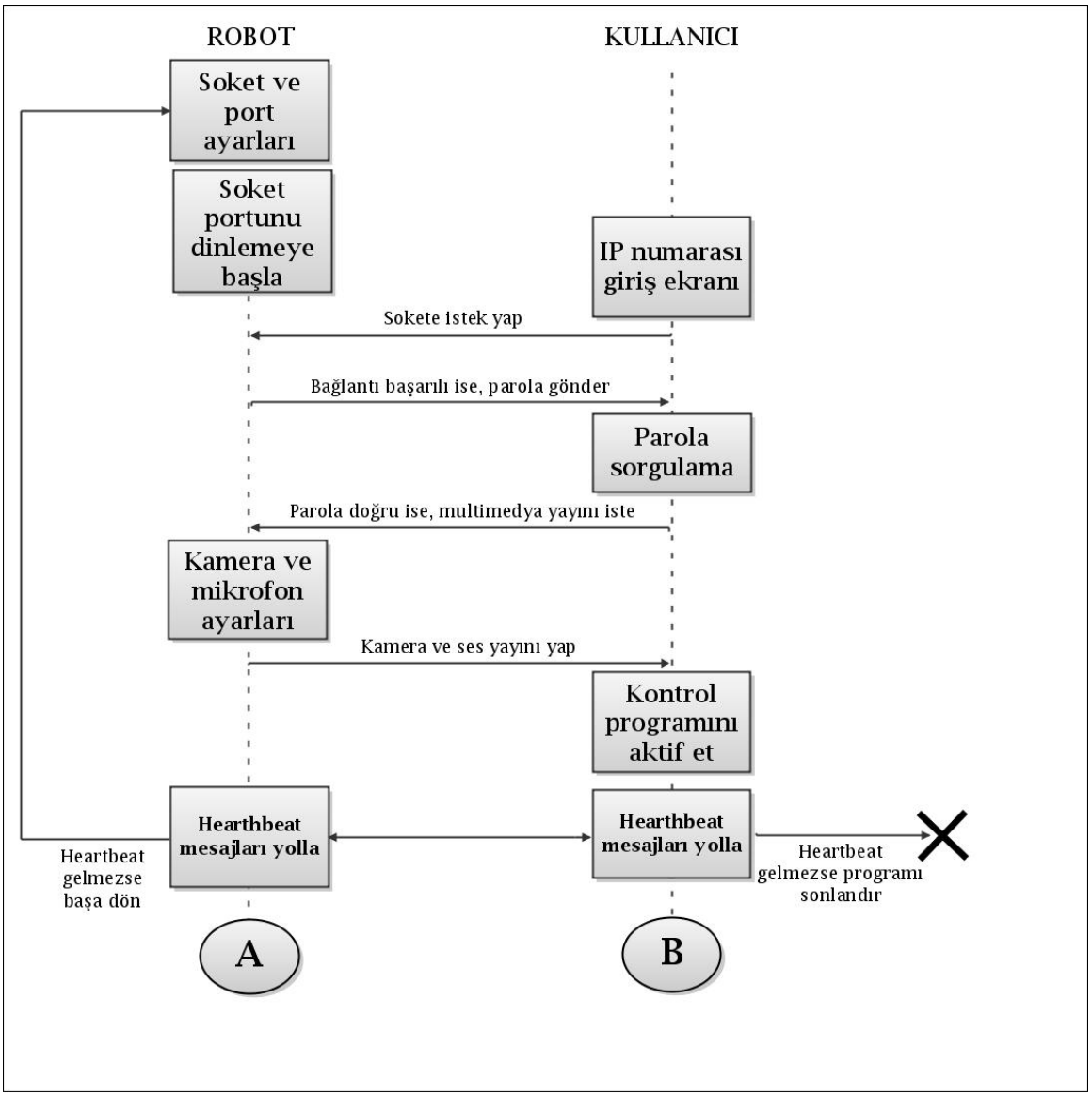
Şekil 5.3'te çalışma prensibi verilen sanal makine gerçekte olmayan bir şeydir. VM komutları, JRE (Java Runtime Environment) tarafından gerçek CPU komutlarına dönüştürülür. Bu şekilde sanal makine uygulama ile platform arasındaki uyumu sağlar. Java programları diğer programlara göre daha yavaş çalışır. Buna karşın bu programların piyasada çok çabuk kabul görmesi Java'ya bugünkü başarısını sağlamıştır.

Sanal makine sayesinde java programlama dili ile yazılmış kullanıcı ve robot kontrol programlarının birçok farklı platformda çalıştırılması mümkün olmaktadır. Bu da mobil keşif robotu için gerçekleştirilen kontrol sisteminin esnek bir yapıya sahip olmasını ve farklı platformlar üzerinde değişikliğe gerek olmadan çalıştırılabilmesini sağlamıştır.

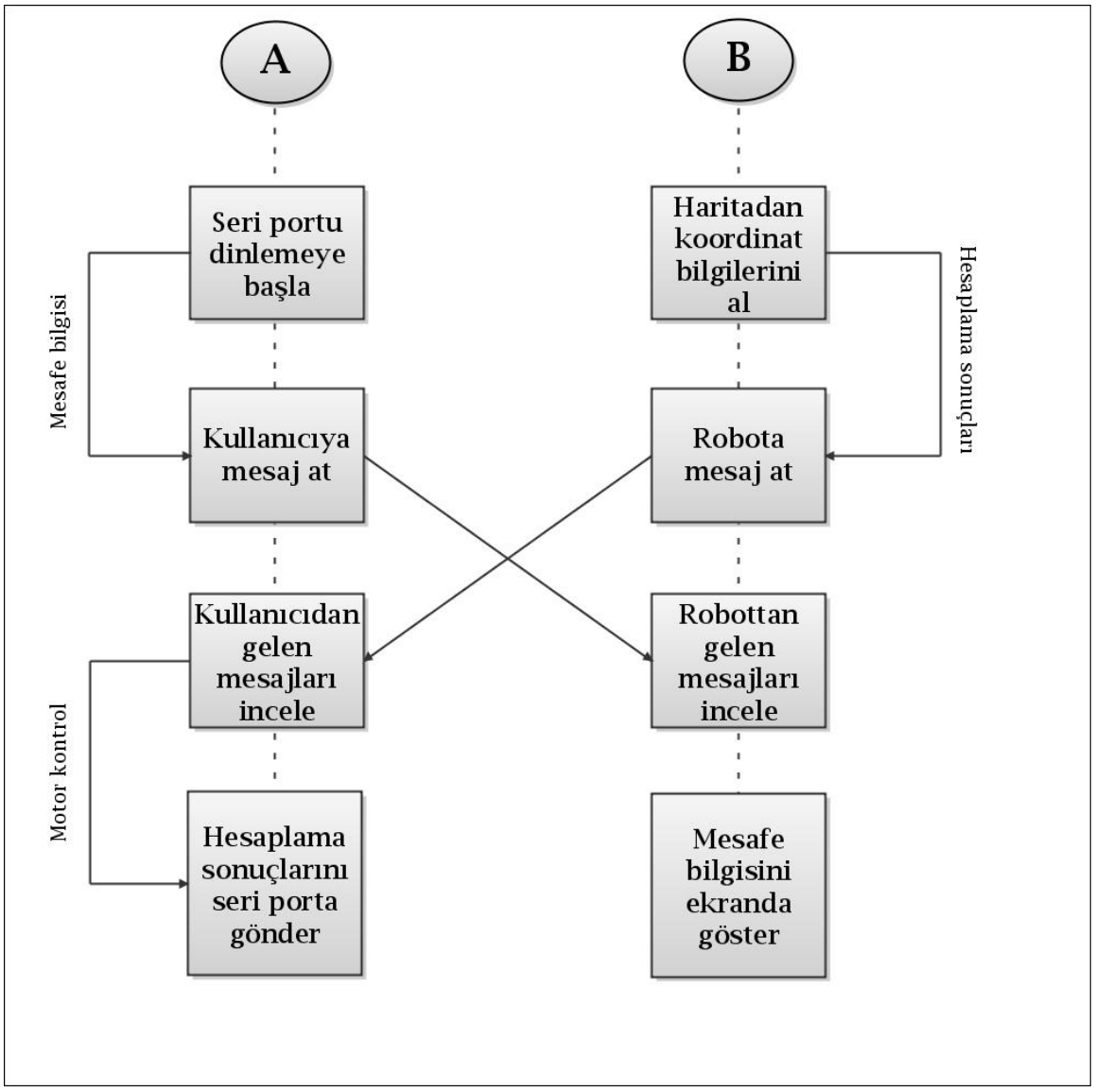
5.3 Robot Kontrol ve Kullanıcı Kontrol Programları

Bu bölümde robot kontrol ve kullanıcı kontrol programlarının çalışma prensipleri bloklar

halinde anlatılmıştır. Şekil 5.4 ve 5.5'te mobil keşif robotu kontrol yazılımlarının blok diyagramları görülmektedir.

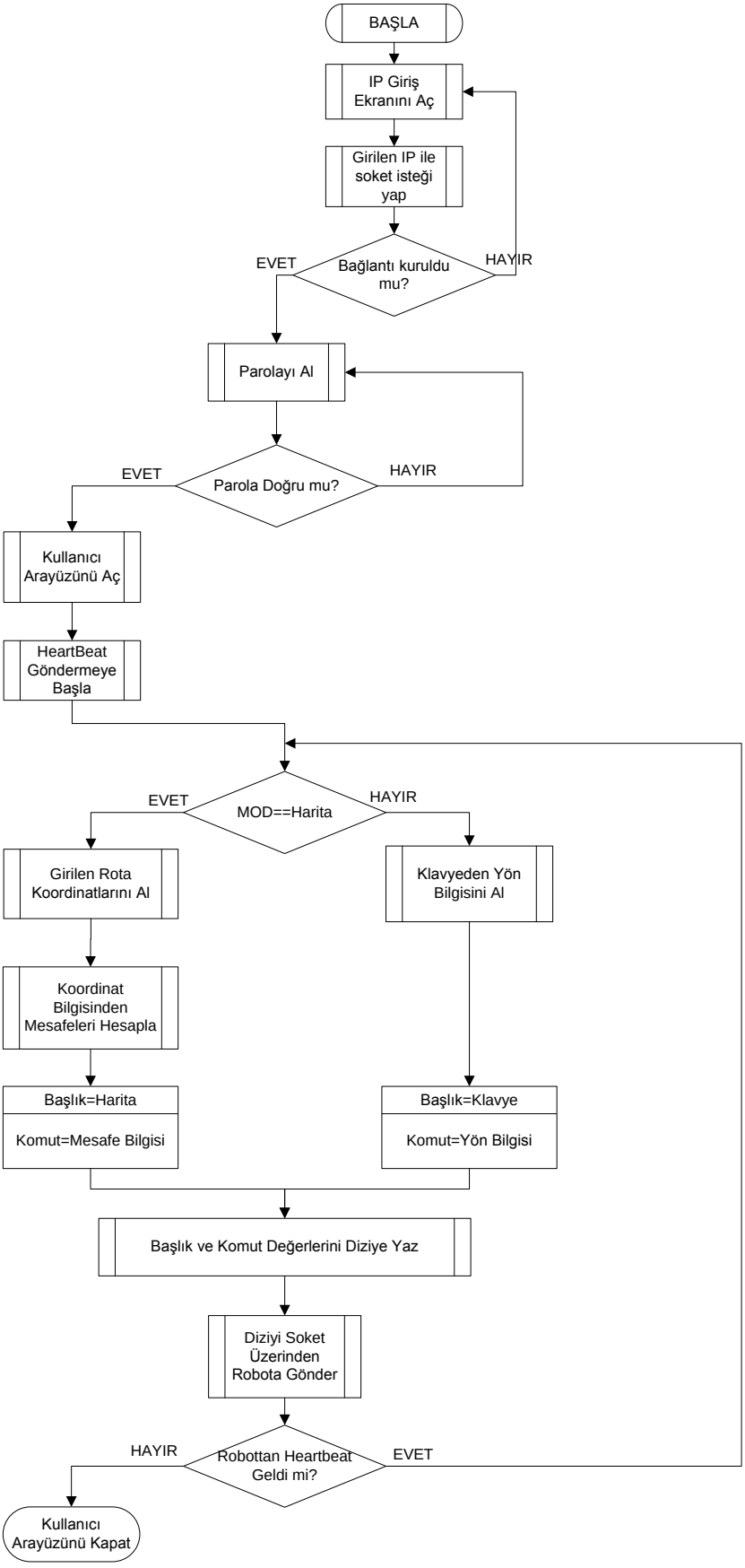


Şekil 5.4 Mobil keşif robotunun yazılımı – Blok Diyagram I

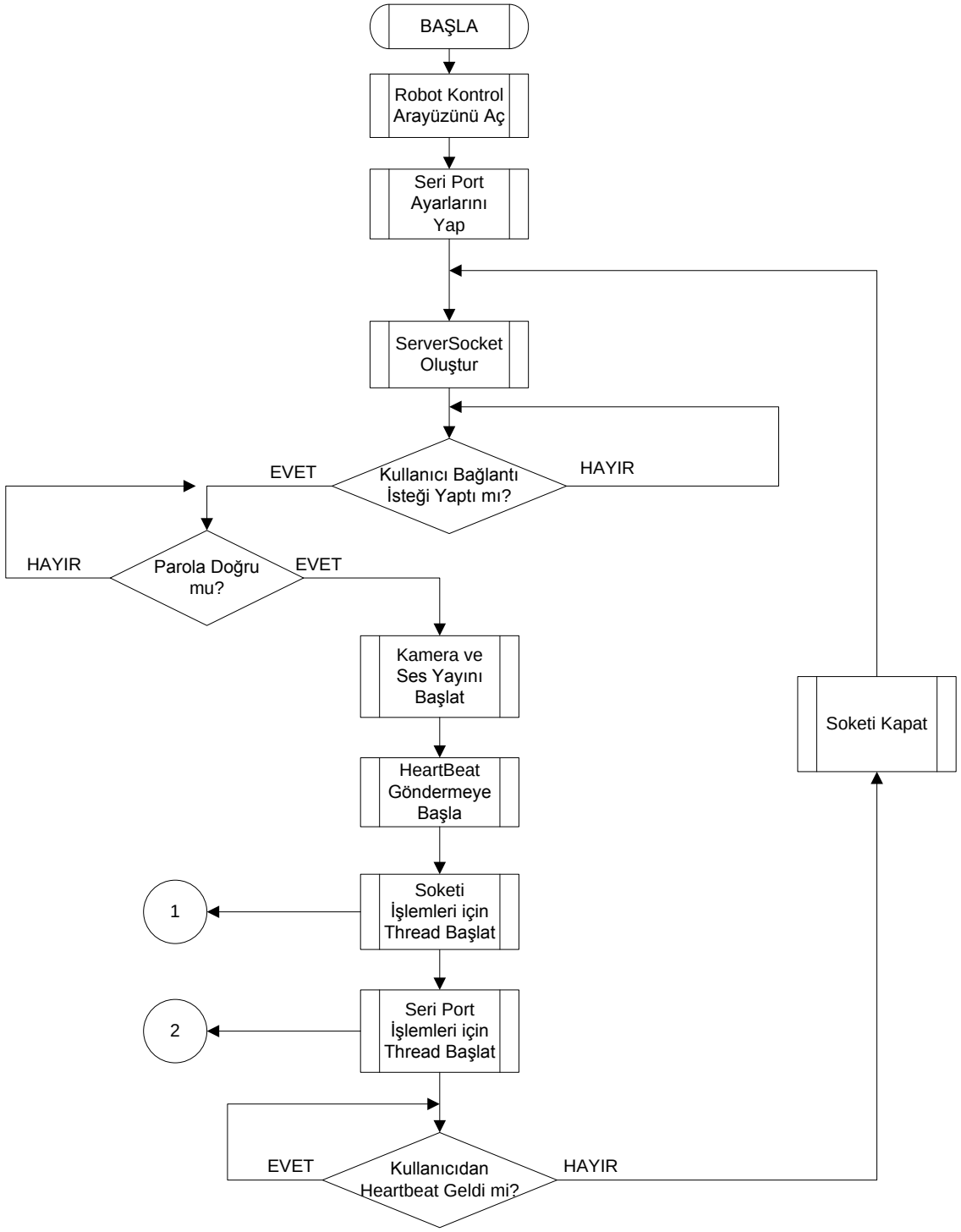


Şekil 5.5 Mobil keşif robotunun yazılımı – Blok Diyagram II

Kullanıcı bilgisayarı üzerinde çalışan kullanıcı kontrol programına ait akış diyagramı Şekil 5.6'da robot üzerinde çalışan ve robotun kontrolünden sorumlu olan kontrol programına ait akış diyagramı ise Şekil 5.7'de görülmektedir.



Şekil 5.6 Kullanıcı kontrol programı akış diyagramı



Şekil 5.7 Robot kontrol programı akış diyagramı

5.3.1 Soket ve Port Ayarları

Robot kontrol programı çalıştığında yapılan ilk iş, kullanıcı kontrol programının bağlantı kurmasını sağlamak amacıyla bir soket yaratılması ve kullanıcıdan gelen hareket bilgileri işlendikten sonra hesaplama sonuçlarının elektronik kontrol kartlarına gönderilmesini

sağlamak için seri port ayarlarının yapılmasıdır.

Soketler taşıma katmanında network programlama için arayüz sağlarlar. Soket kullanılarak network haberleşmesi I/O dosya açma ile çok benzerdir. Soket iki makine ya da işlemci arasındaki bağlantı terminalini temsil eder. Java ile diğer makineyle bağlantı kurabilmek için soket yaratılması gerekmektedir.

Hem istemciler hem de sunucular tarafından kullanılan Java'nın Socket sınıfı uzaktaki bir makineye bağlanma, veri gönderme, veri alma ve bir bağlantıyı kapama işlemlerine karşılık gelen metotlara sahiptir. Bir yerel porta bağlanma, gelen veriyi dinleme ve istemcilerden gelen bağlantıları kabul etme işlemleri, istemcilerin bağlantı yapmalarını beklemeleri için, sadece sunucular tarafında gerekmektedir. Socket sınıfı genel veri iletişimi için kullanılabilecek bir yapıdır. ServerSocket sınıfı ise gelen istekleri kabul eden ve onlara hizmet verecek olan soketleri oluşturan bir yapıdır.

Soket tabanlı haberleşmede sunucu standart bir port numarasından bağlantı isteği bekler. Kendisine bağlı bulunan istemcilere hizmet verebilmek ve aynı zamanda yeni istekleri de dinleyebilmek için her bir bağlantı için yeni bir socket nesnesi oluşturur. İstemci tarafında ise, istemci sunucunun çalıştığı makinenin adını veya IP numarasını ve sunucunun bağlı olduğu port numarasını bilir. Bir bağlantı isteği yapmak için, istemci sunucunun makinesini (IP veya adı) ve port numarasını belirterek, sunucu ile bir bağlantı talebinde bulunur. Sunucu çalışıyor ve istenen hizmette belirtilen port numarasında veriliyorsa, sunucu bağlantı isteğini kabul eder. Kabulden hemen sonra, sunucu istemci ile veri iletişimi için farklı bir porta bağlı bir soket açar. Sunucu bağlanan istemcinin isteklerine bakabilmesi ve aynı zamanda bağlantı isteklerini dinlediği orijinal soketini dinlemeye devam edebilmesi için yeni bir sokete ihtiyaç duyar. İstemci tarafında eğer bağlantı kabul edilmişse bir istemci soketi istemci makinesinde oluşturulur ve istemci sunucu ile haberleşmesinde bu soketi kullanır.

İstemci ve sunucu yukarıdaki adımlardan sonra soketlerine yazarak ve okuyarak haberleşebilir. İstemci ve sunucunun bir protokol üzerinde anlaşmaları, yani soketler üzerinden aktaracakları bilginin dilini belirlemeleri gerekmektedir.

Aşağıda, java programlama dili kullanılarak bir sunucu soket yaratan ve bağlantı için kullanıcı beklemeye başlayan kod bloğu verilmiştir. Buna göre ilk olarak 4040 portunu dinleyen bir sunucu soket oluşturulmuş ve accept metodu çağırılarak kullanıcı beklenmeye başlanmıştır. Soketin yaratılma aşamasında herhangi bir hata ile karşılaşılması durumunda ise ekrana hata basılmaktadır.

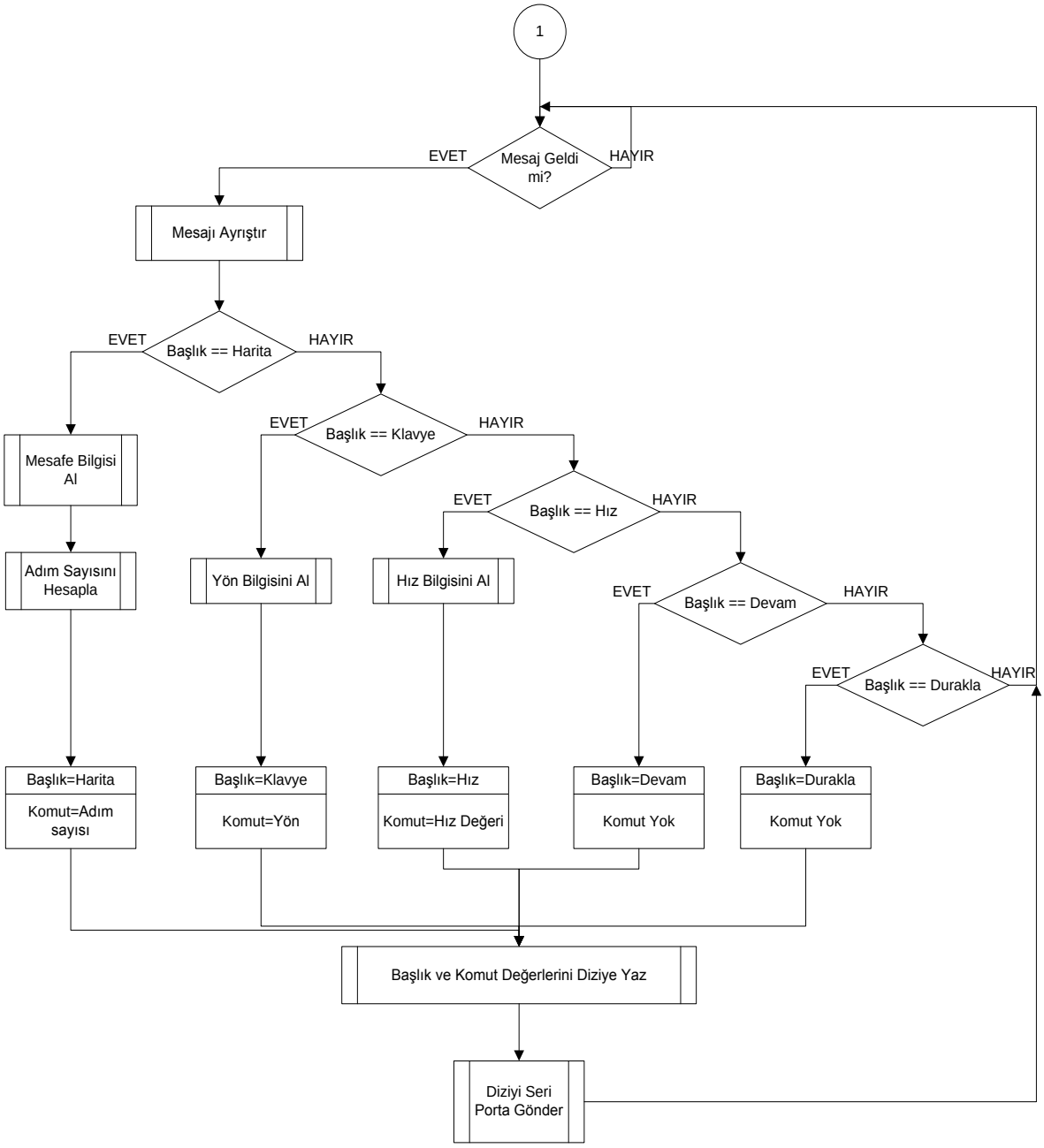
```
try{
   Main.sersoc = new ServerSocket(4040);
   log("Kullanici Bekleniyor.");
   Main.soc = Main.sersoc.accept();

}catch(Exception e){

   log("Socket hatasi.");

   e.printStackTrace();

}
```



Şekil 5.8 Soket alt programı akış diyagramı

Robot kontrol programı üzerinde koşan ve kullanıcı tarafından gönderilen kontrol mesajlarının alınması, ayrıştırılması ve gerekli aksiyonların alınmasını sağlayan soket alt programının akış diyagramı Şekil 5.8'de görülmektedir.

5.3.2 IP Numarası Giriş Ekranı

Kullanıcı programı çalıştığında yapılan ilk iş kullanıcıdan bağlantı kurulacak robotun IP numarasının istenmesidir. Kullanıcının IP numarasını girdikten sonra "Tamam" butonuna basarak bağlantıyı başlatması istenmektedir. "Tamam" butonuna basıldıktan sonra kullanıcının herhangi bir IP numarası girip girmediğinin kontrolü yapılmaktadır. "IP No" alanına herhangi bir giriş yapılmamış olması durumunda ekranda bir hata mesajı gözükmekte ve kullanıcının geçerli bir giriş yapması istenmektedir. Kullanıcı IP numarasını girdikten sonra robot ile bağlantı kurulmaya çalışılmaktadır. Bağlantının başarısız olması durumunda ekranda "Bağlantı kurulamadı!" şeklinde bir hata mesajı görülmekte, başarılı olması durumunda ise parola sorgulama ekranı açılmaktadır.

```
try{
// Girilen IP numarasının 4040 nolu portuna bağlan
  Main.soc = new Socket(IPNumarasi,4040);
}
catch(Exception e){
   // Eğer bağlantı kurulamazsa hata mesajı ver.
   HataMesaji hata = new HataMesaji("Baglanti kurulamadi!",AcilisEkrani.ipsorgu);
   hata.setVisible(true);
   // IP girme alanını temizle.
   jTextField1.setText("");
   bek.setVisible(false);
   return;
}
  Main.par = new Parola(); // Parola isteme ekranını oluşturuyoruz
```

5.3.3 Parola Sorgulama

Robot ile bağlantı kurulduktan sonra kontrol programının açılabilmesi için kullanıcının parola girmesi gerekmektedir. Bunun için "Parola Sorgulama" ekranı açılmakta ve kullanıcının parolayı bu ekrana girmesi istenmektedir. Girilecek olan parolanın doğru olup olmadığının

anlaşılması için iki farklı karşılaştırma yapılmaktadır. Bunlardan bir tanesi robot ile bağlantı kurulur kurulmaz robot tarafından gönderilen parola diğeri ise kullanıcı programında standart olarak bulunan ve kaynak kod dışında değişiklik yapılmasının mümkün olmadığı bir paroladır. Girilen parolanın bu iki paroladan biriyle aynı olmaması halinde bir hata mesajı verilmekte ve kullanıcının doğru parolayı girmesi istenmektedir. Parolanın doğru olması halinde ise kullanıcı kontrol ekranı açılmakta ve robotun multimedya yayını yapması istenmektedir.

```
if (parola.equals(gelenParola)|| parola.equals("admin")){ // Admin parolamız var
    Main.kont.setVisible(true); // Parola doğru ise kullanıcı ekranını açıyoruz
    Iste istek = new Iste(); // Multimedya yayını yapılmasını istiyoruz
    Thread th=new Thread(istek);
    th.start();
    this.setVisible(false);
    this.setEnabled(false);
}else{
    HataMesaji hata = new HataMesaji("Yanlis Parola Girdiniz!",this);
    hata.setVisible(true);
    jPasswordField1.setText(""); // Parola yanlış ise tekrar parola istiyoruz
    this.setEnabled(false);
    return;
}
```

5.3.4 Görüntü ve Ses Aktarımı

Bu projede görüntü ve sesin robot programından kullanıcı programına aktarılması java içerisindeki JMF paketi ile gerçekleştirilmiştir.

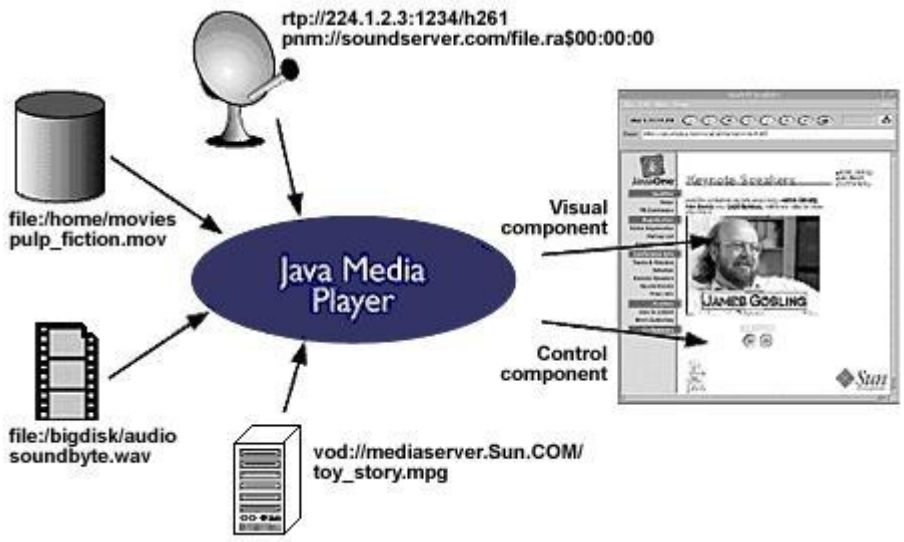
5.3.4.1 JMF Modeli

JMF, java programlarında medya akışını işlemek için bir kullanılan yapıdır. JMF, Java 2 standart platformunda seçmeli pakettir. Java Medya Framework (JMF) zaman tabanlı medyayı java uygulamaları ve appletleriyle birleştiren arayüz uygulama programıdır. JMF birleşmiş mimari ve mesajlaşma protokolü sağlar. JMF'in özelliklerini aşağıdaki gibi sıralayabiliriz:

- Multimedya içeriklerini gösterir.
- Mikrofon ve kameradan veriyi yakalar.
- İnternet üzerinden gerçek zamanlı medya akışı sağlar.
- Medyayı işler (medya formatını değiştirir, özel etkiler katar).
- Dosyaya kaydeder.

JMF medyayı işleyen tüketici elektronik endüstrisi ile aynı modeli kullanır. JMF modeline göre medyanın yaşam döngüsü medya kaynağından başlar, medya alıcı ile biter. Arada medya işleyiciler ile işlenir. Medya kaynağı, yakalama aleti (mikrofon veya kamera) veya network üzerinden gerçek zamanlı medya akışı olabilir. Medya işleme donanım ile gerçekleştirilebilir fakat çoğunlukla yazılım ile yapılır.

Bir görüntü ya da ses dosyası, bir sunucu ya da medya akış kaynağından Şekil 5.9'da görüldüğü gibi Java Media Player aracılığıyla alınarak bir web sayfası ya da bir java programı içerisinde izlenebilir.



Şekil 5.9 JMF ile görüntü ve ses aktarımı

JMF Real-time Transport Protocol (RTP) kullanarak medya akışını ve RTP oturumlarının yönetimini destekler. Karşılıklı farklı medya veri tipleri, protokolleri ve teslim mekanizmasını ölçekler. İsteğe göre uyarlamaya ve genişletmeye uyumlu bir mimari sağlar. Teknoloji

sağlayıcıları JMF'i ilave medya formatını desteklemesi için genişletebilir. Donanım hızlandırıcı olarak kullanılan yüksek performans özel medya player uygulaması veya kodları JMF ile entegre edilebilir ve tanınabilir.

JMF API, JMF nesnelerinin özelliklerinin kontrol edilebilmesi için bazı arayüzler sağlar. Player, Processor, DataSource gibi. Örneğin GainControl nesnesi oynatıcının ses denetimini kontrol eder. Manager sınıfı DataSource, DataSink, Player, Processor, cloneableDataSource, ve MergedDataSource gibi JMF nesnelerini yaratmak için kullanılır.

Ses ve görüntü yakalama aletleri, kameralar ve ses kartlarını içerir. Bütün ses ve görüntü yakalama aletleri CaptureDeviceManager ile saklanır. JMF ile onları kullanmak için, CaptureDeviceManagerı verilen çıkış formatında desteklenen ses ve görüntü yakalama aletlerinin listesi için sorgulanabilir.

Bütün multimedya içerikleri çeşitli standart formatlar kullanılarak sıkıştırılmış biçimde saklanır. Her format temel olarak medya kodlama kullanan metodu tanımlar. Bu yüzden multimedya içeriklerinin formatını tanımlayan sınıfa gereksinim duyulur. JMF uygun medya format özelliklerini belirten Format sınıfını tanımlar. Format sınıfı AudioFormat ve VideoFormat sınıfları olarak özelleştirilir.

5.3.4.2 Kamera ve Ses Yayını

Kullanıcı programı robot ile soket bağlantısı kurduktan sonra, robot programı soketi dinlemeye başlamaktadır. Kullanıcı soket üzerinden robota birtakım istekler göndermekte ve robot da gelen isteklerin içeriklerine göre birtakım aksiyonlar almaktadır. Soket üzerinden gelen mesajlar metin bazlı mesajlar olup bu metinlerin ilk kelimeleri mesajın tamamı veya varsa devamı hakkında bilgi içermektedir. Kullanıcının doğru parolayı girmesi halinde yapılan ilk iş robotun multimedya yayını yapmasını istemesi olduğundan bu istek "kamera" ve "ses" şeklinde iki mesaj göndererek yapılmaktadır. Robot tarafından alınan mesajlar ayrıştırılmakta ve gelen mesajların içinde ilk kelimesi "kamera" veya "ses" olan bir mesaj olması halinde multimedya yayını yapılmaya başlamaktadır.

```
if (stringdizisi[0].equals("ses")){ // Gelen bilgi ses ise ses yayını yap
        ses sound = new ses();
        Thread th4=new Thread(sound);
        th4.start();
    }else if (stringdizisi[0].equals("kamera")){ // Kamera yayını yap
```

```
        kamera cam = new kamera();
        Thread th3=new Thread(cam);
        th3.start();
    }
```

Kamera ve ses yayınları java programlama dilindeki AVTransmit2 sınıfı kullanılarak yapılmakta ve yayının başarılı şekilde yapılmaya başlanması durumunda soket üzerinden "kamera on" ve "ses on" mesajları gönderilmektedir. Kullanıcı kontrol programı ancak bu mesajları aldıktan sonra "Kamera Aç" ve "Ses Aç" butonlarını aktif hale getirmektedir. "kamera off" veya "ses off" mesajlarının gelmesi ise multimedya yayını yapılamadığını ve "Kamera Aç" ya da "Ses Aç" butonlarının aktif hale getirilmemesi gerektiğini bildirmektedir.

```
Main.kon.log("Kamera Yayini Yapiliyor... ");

Main.transmit = new AVTransmit2(new MediaLocator("vfw://0"),
String result = Main.transmit.start();// Yayını başlat
if (result != null) {
    Main.kon.log("Kamera Hatasi : " + result);
    Main.kon.mesajGonder("kamera off ");
    return;
}
Main.kon.log("Kamera Iletimi Basladi...");
Main.kon.mesajGonder("kamera on ");
Main.kon.log("Ses Yayini Yapiliyor... ");

Main.transmit2 = new AVTransmit2(new MediaLocator("dsound://"),
String result2 = Main.transmit2.start(); // Yayını başlat
if (result2 != null) {
    Main.kon.log("Ses Hatasi : " + result2);
    Main.kon.mesajGonder("ses off ");
    return;
}
```

```
Main.kon.log("Ses Iletimi Basladi...");
Main.kon.mesajGonder("ses on ");
```

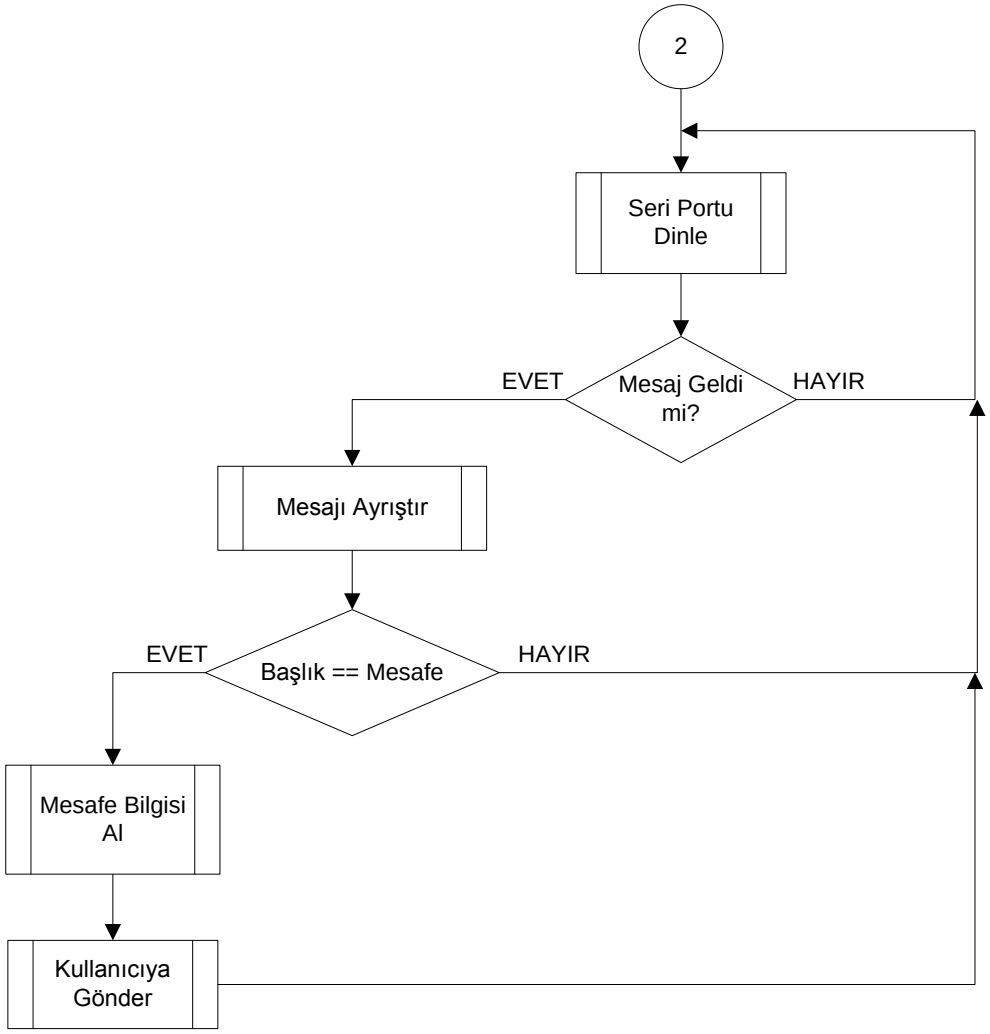
5.3.5 Heartbeat Mesajları

Heartbeat mesajları kullanıcı kontrol ve robot kontrol programlarının birbirlerinin durumlarından haberdar olmak için gönderdikleri metin bazlı mesajlardır. Her iki program da beşer saniye aralıklarla "handshake" şeklinde mesajlar göndermekte ve on saniye boyunca karşı taraftan mesaj gelmemesi durumunda gerekli aksiyonları almaktadırlar. Buna göre kullanıcı programına on saniye boyunca herhangi bir mesaj gelmemesi durumunda "Bağlantı koptu!" şeklinde bir hata mesajı verilmekte ve program kapatılmaktadır. Bu durumda program kullanıcı tarafından yeniden çalıştırılarak robot ile yeni bir bağlantı kurulmalıdır. Robot programına on saniye boyunca herhangi bir mesaj gelmemesi durumunda ise multimedya yayını durdurulmakta ve soket yeniden yaratılıp kullanıcı tarafından yeni bir bağlantı kurulması beklenmektedir.

5.3.6 Seri Porttan Okuma ve Yazma İşlemleri

Robot kontrol programının en önemli görevlerinden bir tanesi ultrasonik mesafe ölçüm kartının seri port üzerinden gönderdiği mesafe bilgisinin alınıp soket üzerinden metin bazlı olarak kullanıcı kontrol programına iletilmesi ve kullanıcı kontrol programının gönderdiği metin bazlı hareket bilgilerinin alınıp işlenerek yapılan hesaplama sonuçlarının yine seri port üzerinden motor kontrol kartına gönderilmesidir.

Seri port okuma ve yazma işlemleri java programlama dilinin SimpleRead ve SimpleWrite sınıfları ile gerçekleştirilmektedir. Buna göre seri port devamlı olarak dinlenmekte ve herhangi bir mesaj geldiğinde ilgili EventHandler çağırılarak gelen mesaj okunmaktadır. Okunan mesafe bilgisi başına "mesafe " kelimesi eklenerek soket üzerinden kullanıcı kontrol programına gönderilmekte ve kullanıcı kontrol programı gelen mesajın ilk kelimesinin "mesafe " olması halinde ikinci kelimeyi mesafe bilgisi olarak kullanıcı kontrol programının ekranında göstermektedir. Bu işlemleri gerçekleştiren seri port alt programına ait akış diyagramı Şekil 5.10'da görülmektedir.



Şekil 5.10 Seri port alt programı akış diyagramı

Seri porttan okuma ve yazma işlemlerini gerçekleştiren alt programa ait java kodunun bir parçası aşağıda verilmiştir.

```
switch (event.getEventType()) {
        case SerialPortEvent.DATA_AVAILABLE:
                byte[] readBuffer = new byte[20];
                try {
                        while (inputStream.available() > 0) {
                        int numBytes  = inputStream.read(readBuffer);
                }
                Integer x = new Integer(readBuffer[0] & 0xFF);
                mesafe = x;
```

```
            Main.kon.log(x.toString());
            mesaj.mesajGonder("mesafe "+x.toString());
```

Kullanıcı kontrol programından gönderilen hareket bilgileri “hareket”, “durakla”, “git” ve “klavye” anahtar kelimelerinden biri ile başlayan mesajlar şeklinde iletilmektedir. Gelen mesajın ilk kelimesinin “hareket” olması gelen mesajın devamının haritadan alınan hareket bilgilerini içerdiğini, “durakla” olması robotun hareketini duraklatması gerektiğini, “git” olması robotun harekete devam etmesi gerektiğini, “klavye” olması gelen mesajın devamının kullanıcının klavyesinden okunan yön bilgilerini içerdiğini göstermektedir.

```
    if (stringdizisi[0].equals("hareket")){ // Harita bilgisi geldi
       int deger[]=new int[11];
       double double_deger[]=new double[11];
       byte komut[]=new byte[26];
       komut[0]='H';
       if (stringdizisi.length >= 11){ // Aldigimiz degerleri ekrana yazıyoruz
          Main.kon.jTextField1.setText(stringdizisi[1]);
          Main.kon.jTextField2.setText(stringdizisi[2]);
          Main.kon.jTextField3.setText(stringdizisi[3]);
          Main.kon.jTextField4.setText(stringdizisi[4]);
          Main.kon.jTextField5.setText(stringdizisi[5]);
          Main.kon.jTextField6.setText(stringdizisi[6]);
          Main.kon.jTextField7.setText(stringdizisi[7]);
          Main.kon.jTextField8.setText(stringdizisi[8]);
          Main.kon.jTextField9.setText(stringdizisi[9]);
          Main.kon.jTextField10.setText(stringdizisi[10]);
          for (int i=1;i<11;i++){
             double_deger[i]=((Double.parseDouble(stringdizisi[i]))*10);
          }
          // Aldığımız double bilgiyi adıma çeviriyoruz
          for (int i=1;i<11;i++){
```

```
            deger[i]= ((int) (double_deger[i]*2));  // Ileri gidiş adımları
            Main.kon.log(new Integer(deger[i]).toString()+" adim atilmali");
        }
        for (int i=2;i<11;i=i+2){
            deger[i]= ((int) (double_deger[i]/0.72));   // dönüş adımları
            Main.kon.log(new Integer(deger[i]).toString()+" adim donulmeli");
        }
        for (int i=1;i<6;i++){
            int index1 = i*2;
            int index2 = ((5*index1)-8)/2;
            if (deger[index1] > 0){
                komut[index2]='-';
            }else{
                komut[index2]='+';
                deger[index1]=Math.abs(deger[index1]);
            }
            komut[index2+1]=(byte)(deger[index1]/255);
            komut[index2+2]=(byte)(deger[index1]-((deger[index1]/255)*255));
            komut[index2+3]=(byte)(deger[index1-1]/255);
            komut[index2+4]=(byte)(deger[index1-1]-((deger[index1-1]/255)*255));
        }
        System.out.println(komut[4] +" " +(komut[5]& 0xFF) + " adim");
        SimpleWrite.robot_komut('M',komut); // Hareket bilgisi donanima gonderildi
    }
}else if (stringdizisi[0].equals("durakla")){ // Durakla geldi ise motorlari durdur
    byte b[]={'D'};
    SimpleWrite.robot_komut('M',b); //MD
}else if (stringdizisi[0].equals("git")){ // Git ise motorlari tekrar serbest birak
    byte b[]={'G'};
    SimpleWrite.robot_komut('M',b); //MG
}else if (stringdizisi[0].equals("klavye")){ // Klaveye modundan bilgi geldiyse
```

```
        byte b[]= new byte[2];
        b[0]='K';
        if (stringdizisi[1].equals("yukari")) { b[1]='U'; Main.yon = "yukari";} //MKU
        else if (stringdizisi[1].equals("asagi")) { b[1]='D';  Main.yon = "asagi";} //MKD
        else if (stringdizisi[1].equals("sola")) { b[1]='R';  Main.yon = "sola";} //MKL
        else if (stringdizisi[1].equals("saga")) { b[1]='L';  Main.yon = "saga"; } //MKR
        else if (stringdizisi[1].equals("dur")) { b[1]='S'; Main.mod="klavye";} //MKS
      }else if (stringdizisi[0].equals("harita")){ // Harita moduna gecildi
        if (stringdizisi[1].equals("modu")) {
          byte b[]={'X'};
          SimpleWrite.robot_komut('M',b);  //MX
          Main.mod="harita";
        }
      }
```

Gelen mesajlar buna göre ayrıştırılıp anlamlı değerlere dönüştürüldükten sonra yapılması gereken hesaplamalar yapılarak atılması gereken adım sayıları belirlenmekte ve bu sonuçlar seri port üzerinde motor kontrol kartlarına gönderilmektedir.

```
  public static void robot_komut(char kart, byte komut[]) {
    try {
      outputStream = Main.kon.serialPort.getOutputStream();
    } catch (IOException e) {}
    try {
      Integer uzunluk = new Integer(komut.length+1);
      outputStream.write(messageString.getBytes());
      outputStream.write(uzunluk.byteValue());
      outputStream.write(kart);
      outputStream.write(komut);
    } catch (IOException e) {}
  }
```

5.4 Ultrasonik Mesafe Ölçüm Kartı Yazılımı

Ultrasonik mesafe ölçüm kartı yine Atmel firmasına ait AVR serisi mikrodenetleyici üzerinde C programlama dili kullanılarak yazılan bir programdır. Bu programın görevi SRF05 ultrasonik mesafe algılayıcısının çalışması için gereken uygun işaretlerin üretilmesi, algılayıcının ürettiği işaretleri inceleyerek mesafe hesabının yapılması ve hesaplanan mesafe bilgisinin RS232C protokolü ile uygun biçimde robot bilgisayara gönderilmesinin sağlanmasıdır. Ultrasonik mesafe ölçüm kartı mikrodenetleyici yazılımı ayrıntılı olarak Ek-3'te verilmiştir.

```
if ((bit_is_set(PIND, 6))){ //çıkan kenar görülürse
    rising=ICR1; //Başlangıç zamanını kaydet
    TCCR1B = _BV(CS10); //düşen kenara ayarla
}else{ //düşen kenar görülürse
    falling=ICR1; //zamanı kaydet
    TCCR1B = _BV(CS10)|_BV(ICES1); //çıkan kenara ayarla
    led = falling - rising; // darbe genişliğini hesapla
    led /=60; //santimetreye çevir
    UDR = (uint8_t)led; // sonucu seri porta gönder
}
```

5.5 Adım Motor Sürücü Kartı Yazılımı

Adım motor sürücü kartı Atmel firmasına ait AVR serisi mikrodenetleyici üzerinde C programlama dili kullanılarak yazılan bir programdır. Bu programın görevi robot bilgisayarın seri portu üzerinden gelen bilgilerin mesaj gönderme biçimine uygun olup olmadığının kontrolü ve uygun olması durumunda alınan bilgilerin anlamlı olacak şekilde derlenmesiyle motorların doğru hareketleri yapmasını sağlamaktır. Program bir taraftan motoru kontrol ederken diğer taraftan gelen mesajları toplayıp doğru biçimde işleyecek şekilde tasarlanmıştır. Ayrıca ultrasonik mesafe ölçüm kartından gelen işaretler vasıtasıyla bilgisayardan gelen mesajlardan bağımsız olarak da motor hareketini duraklatabilmektedir. Seri porttan gelen mesajların incelenmesi üç ayrı modda yapılmaktadır. İlk modda seri porttan gelen mesajın ilk karakterinin "B" olup olmadığı kontrol edilir. Robot bilgisayarından gelen bütün mesajlar "B" karakteri ile başlamaktadır. İlk karakterin "B" olması durumunda ikinci moda geçilir. İkinci modda gelen ikinci byte incelenerek mesajın uzunluk bilgisi elde edilmektedir. Çünkü motor kontrol kartına gelen mesajlar aynı uzunlukta değildir. Uzunluk bilgisi alındıktan sonra

üçüncü moda geçilmekte ve bu modda mesajın tamamı alınarak bu bilgiler tampon belleklerde saklanmaktadır. Mesajın tamamı alındıktan sonra tekrar birinci moda geçilerek yeni bir mesaj dizisi beklenmekte ve tamponlardaki bilgiler bir araya getirilerek motorun yapması gereken hareketler kontrol edilmektedir. Motor sürücü kartı mikrodenetleyici yazılımı detaylı olarak Ek-3'te verilmiştir.

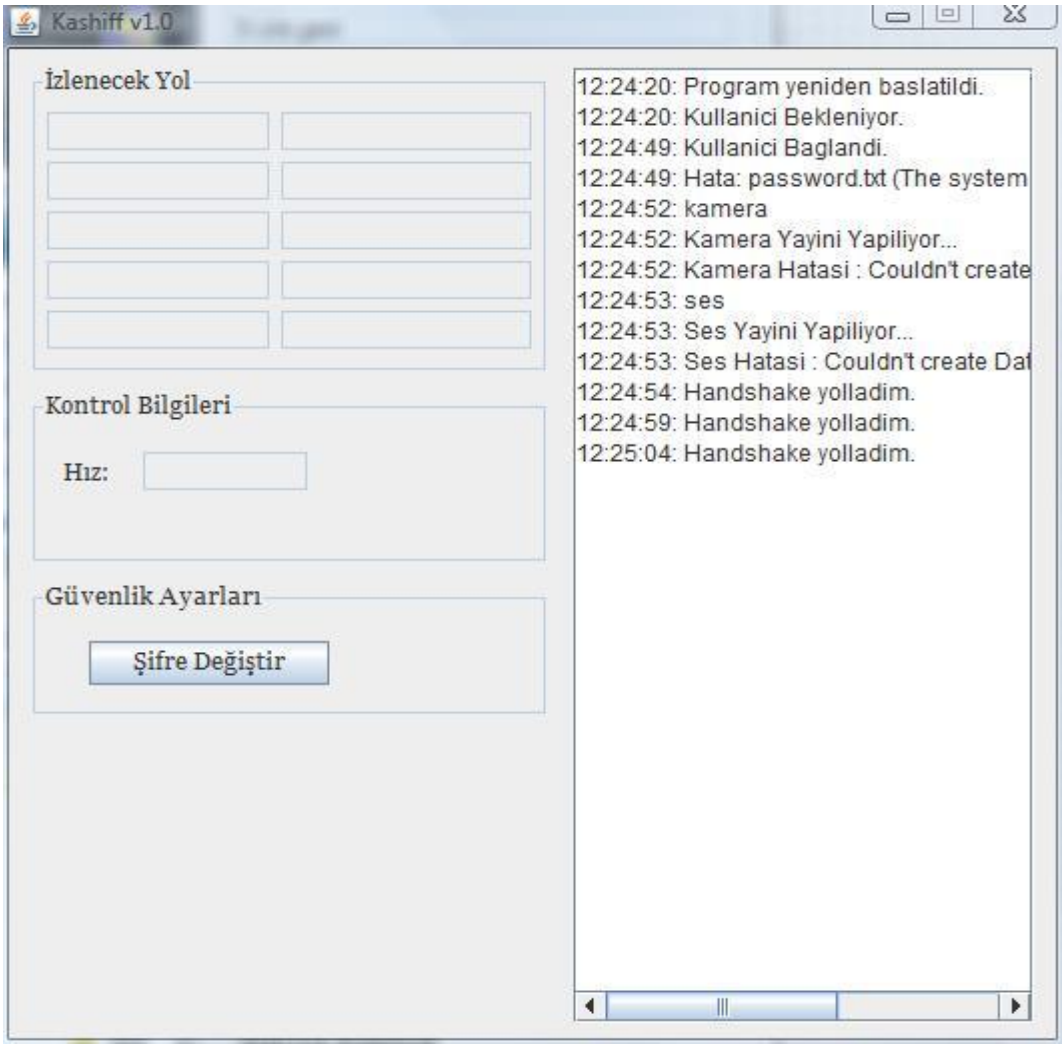
```
uint8_t get;  get = UDR; // Seri porttan veriyi aldık
if (mod==1) //mod1, , "B" ozel karakterini bekle
{
    if (get == 'B'){ // gelen karakter "B" ise
     mod =2; // mod 2'ye geç
    }
} else if (mod == 2){//mod2, uzunluk bilgisini al
    uzunluk=get;
    sayac=0;
    mod=3; // mod 3'e geç
} else if (mod == 3){//mod3, bilgileri tamponlara yaz
    uzunluk=uzunluk-1;
    bilgi[sayac]=get;
    sayac++;
    if (uzunluk == 0){
        mod = 1; //mod 1'e geç
        updated=1;
    }
}
```

6. MOBİL ROBOTUN KULLANIMI

Mobil robot, kullanıcı ve robot bilgisayarları üzerinde çalışan, Java programlama dili kullanılarak geliştirilmiş iki farklı uygulamadan oluşmaktadır. Bu bölümde bu programların kullanımı hakkında bilgi verilmektedir.

6.1 Robot Kontrol Programı Kullanımı

Robot programı robota güç verilerek mini bilgisayarın çalıştırılmasının ardından otomatik olarak çalışmakta ve önceden belirlenen sabit port üzerinden bir soket açmaktadır. Bu aşamadan sonra kullanıcı programından bir bağlantı yapılıncaya kadar beklenmektedir. Kullanıcı programından bir bağlantı yapılması ve doğru parolanın girilmesinin hemen ardından karşılıklı olarak düzenli veri alışverişi başlamaktadır. Şekil 6.1'de programın açılış anındaki ekran görüntüsü verilmiştir.

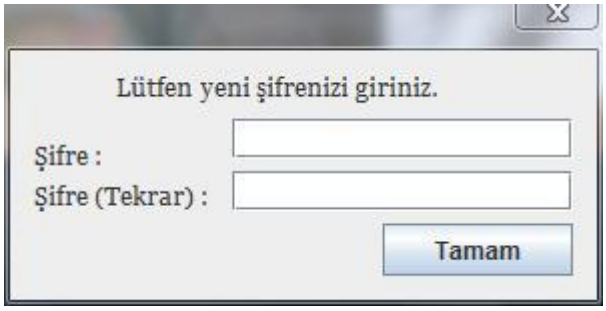


Şekil 6.1 Robot kontrol programı

Kullanıcı tarafından gelen bilgi “Harita Modu” ile çizilen yola ait bir bilgi ise, sağa-sola ve ileri-geri gidiş mesafe bilgileri içeren veri adım motorların atması gereken adım değerlerine çevrilerek elektronik kartların anlayabileceği bir formata dönüştürülmekte ve seri port üzerinden elektronik kartlara gönderilmektedir. Ayrıca bu bilgiler hata ayıklama amaçlı olarak program üzerindeki “İzlenecek Yol” alanında da gösterilmektedir.

Robot kontrol programı tarafından gönderilen hız bilgileri de benzer şekilde işlenerek elektronik kartların anlayabileceği formata dönüştürülmekte ve seri port üzerinden kartlara gönderilmektedir. Hız bilgisi ekrandaki “Hız” alanında da gösterilmektedir.

Robot programı üzerinde bulunan “Şifre Değiştir” butonu kullanıcı tarafından girilmesi gereken şifrenin değiştirilmesi işlemini gerçekleştirmektedir. Butona basıldığı zaman Şekil 6.2’de görülen ekran açılmaktadır. Açılan bu ekranda değiştirilmek istenen şifrenin iki kere yazılması istenmekte ve bu iki şifrenin de aynı olması durumunda şifre değiştirilmektedir.



Şekil 6.2 Şifre değiştirme ekranı

Robot kontrol programı tarafından alınan bilgiler herhangi bir problem olması halinde incelenmesi için programın bilgi ekranında gösterilmektedir. Robot programının yaptığı temel işlerden bir diğeri ise JMF modeli ile ortamdan alınan ses ve görüntü bilgilerinin RTP protokolü üzerinden kontrol programına gönderilmesidir. Kullanıcı ve robot programları arasındaki RTP üzerinden bilgi alışverişinin başladığına dair mesajlar ve hata durumundaki mesajlar da bilgi ekranında gösterilmektedir.

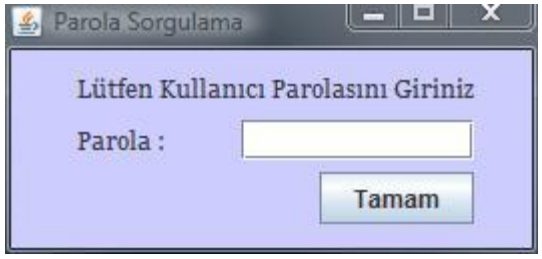
Kontrol programı tarafından gönderilen komut bilgileri belli bir formata sahiptir. Robot programı alınan bilgilerin bu formata sahip olmaması durumunda gelen bilgileri dikkate almamakta ve bununla ilgili hata mesajlarını bilgi ekranına basmaktadır. Doğru formata sahip bilgilerin gelmesi durumunda ise bu bilgiler içeriklerine göre farklı şekillerde işlenerek gerekli aksiyonlar alınmaktadır.

6.2 Kullanıcı Kontrol Programı Kullanımı

Kontrol programını çalıştırmak için gerekli olan JAR dosyasını çalıştırmak suretiyle gelen ilk ekran Şekil 6.3'te görülen IP numarası giriş ekranıdır. Bu ekrandaki alana robotun IP numarası girildikten sonra "Tamam" tuşuna basılarak robot ile bağlantı kurulması sağlanır. Robot üzerindeki mini bilgisayar üzerinde yapılan ayarlara bağlı olarak, robota bağlanmak için kullanılan IP numarası sabit olabileceği gibi dinamik de olabilmektedir.

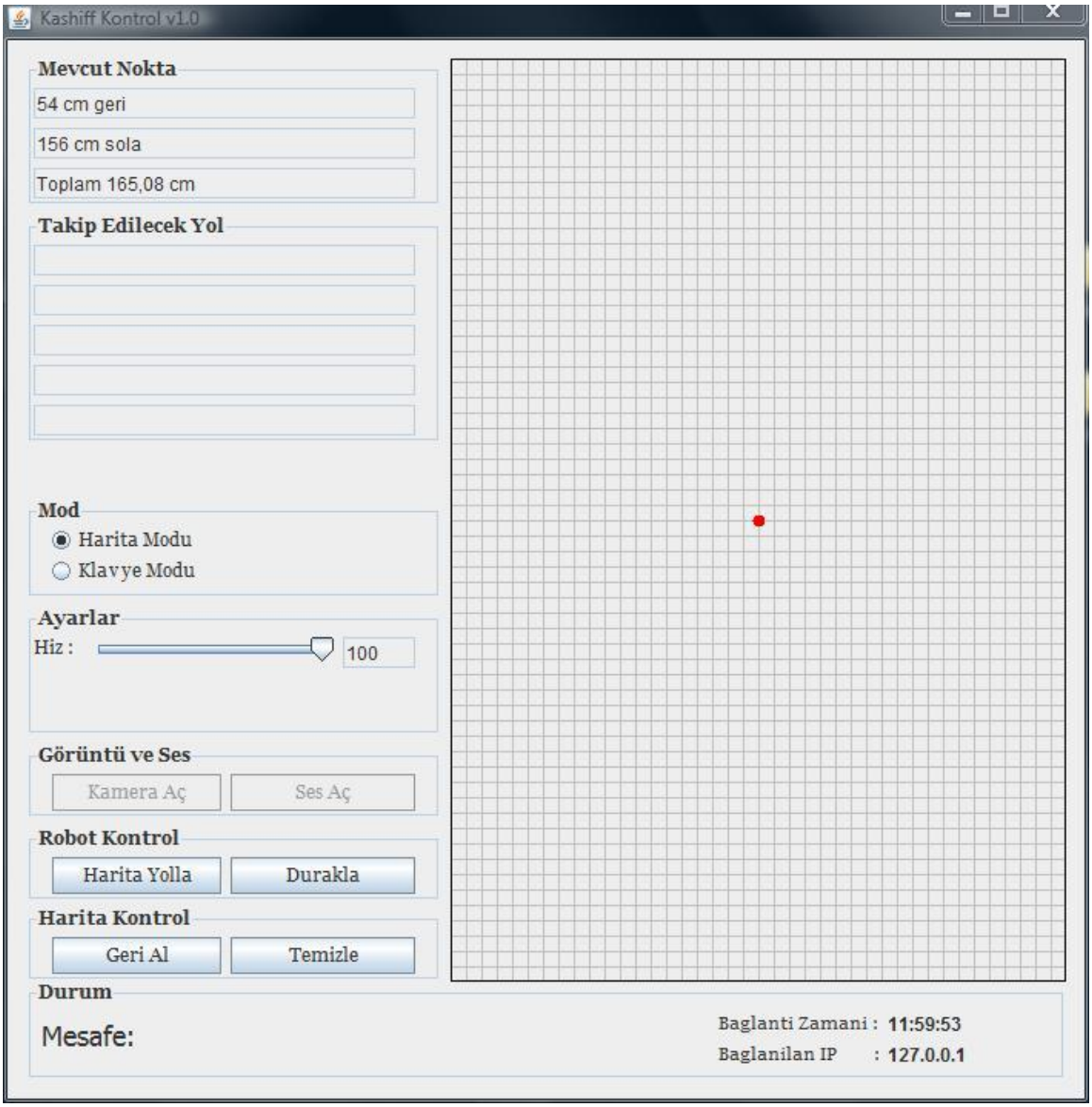
Şekil 6.3 IP numarası giriş ekranı

Robot ile bağlantı kurulması halinde karşımıza çıkacak olan ekran Şekil 6.4'te görülen parola sorgulama ekranıdır. Bu ekrana parolayı girdikten sonra "Tamam" tuşuna basarak şifrenin doğru olması halinde kontrol programının çalışmasını sağlamış oluruz.



Şekil 6.4 Parola sorgulama ekranı

Doğru parola girildikten sonra açılan ekran Şekil 6.5'te görülen robot kontrol ekranıdır. Bu ekran vasıtası ile robotun kontrolü için gerekli bütün komutlar verilebildiği gibi robot tarafından gönderilen ses, görüntü ve mesafe bilgileri de bu ekran aracılığı ile gözlemlenmektedir.



Şekil 6.5 Kontrol ekranı

Kontrol ekranının sağ alt köşesinde bağlantı kurulan robotun IP numarası ve bağlantının yapıldığı zaman bilgileri görülebilmektedir. Sol alt köşesinde ise robotun önündeki engele olan mesafesinin ne kadar olduğu bilgisi santimetre cinsinden verilmektedir.

Kontrol ekranının sol alt bölümünde bulunan "Kamera Aç" ve "Ses Aç" butonları kullanılarak kamera ve ses ekranlarının açılması sağlanabilir. Kontrol programı açıldığında kamera ve ses ekranları kapalı olarak gelmektedir.

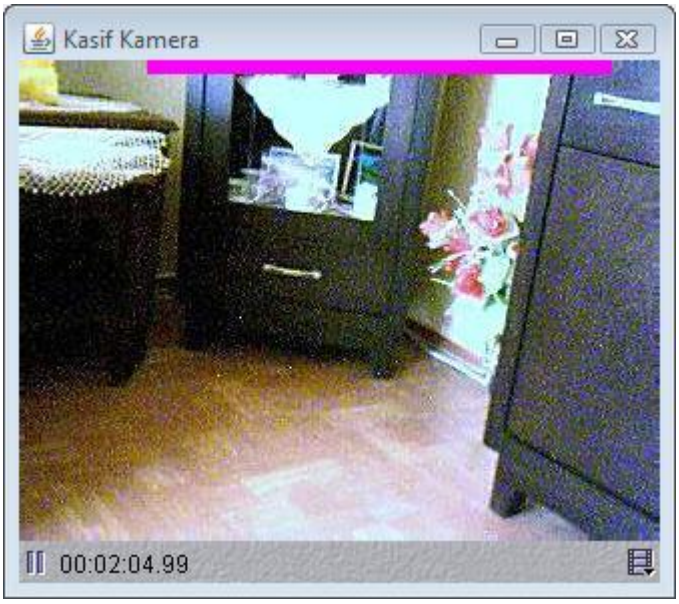
"Ses Aç" butonuna basıldığında Şekil 6.6'da görülen ses ekranı aktif hale gelmektedir. Bu ekran, robot tarafında kullanılan JMF yapısı ile ortamdan alınan seslerin RTP protokolü ile kullanıcı tarafına aktarılmasını ve bu seslerin kullanıcıya dinletilmesini sağlamaktadır. Ses

ekranı üzerinde bulunan ayarlar sayesinde gelen sesin kısılması veya arttırılması gibi işlemler yapılabilmektedir. Ses ekranı açık olduğu sürece ses butonu inaktif durumdadır. Ekran kapatıldığında ise buton tekrar aktif hale gelmektedir.

Şekil 6.6 Ses ekranı

"Kamera Aç" butonuna basıldığında Şekil 6.7'de görülen kamera ekranı aktif hale gelmektedir. Bu ekran, robot tarafında kullanılan JMF yapısı ile ortamdan alınan görüntünün RTP protokolü ile kullanıcı tarafına aktarılmasını ve bu görüntünün kullanıcı tarafında izlenebilmesini sağlamaktadır. Kamera ekranı açıldığında gelen görüntü tam ekran görüntüsü değildir. Kontrol programının görünür halde olması için küçük bir ekran şeklinde ayarlanmıştır. Fakat bu ekran tam ekran da dahil olmak üzere farklı büyüklüklere ayarlanabilmektedir.

Kamera ekranı açık olduğu sürece kamera butonu inaktif durumdadır. Ekran kapatıldığında ise buton tekrar aktif hale gelmektedir.



Şekil 6.7 Kamera ekranı

Kontrol ekranının sağ tarafında görülen ızgara şeklindeki alan “Harita Modu” kullanılırken robotun takip etmesini istediğimiz yolu çizeceğimiz sanal rotadır. Bu alan “Klavye Modu” kullanılırken ise klavyenin hangi tuşlarına bastığımızı göstermek için kullanılır.

Ekranın sol tarafında bulunan alanlar “Harita Modu” kullanılırken yardımcı olacak konum ve yol bilgilerini yazılı olarak göstermek için kullanılmaktadır. Ayrıca mod seçimi yapmak için kullanılan alan da bu kısımda bulunmaktadır. Robot kontrolündeki önemli parametrelerden biri olan hız değeri de bu kısımdaki ayarlar alanından değiştirilebilmektedir.

Kontrol programı ile robot üzerinde çalışan program arasında devamlı surette bir veri alışverişi olmaktadır. Bu veri alışverişi sayesinde programlar arasında oluşabilecek kopuklukların fark edilmesi ve gerekli önlemlerin alınabilmesi sağlanabilmektedir. İki program arasında oluşacak herhangi bir kopukluk neticesinde kontrol programı kullanıcıya bilgi verdikten sonra kapatılmakta, robot üzerinde çalışan program ise kendi kendine yeniden başlamaktadır.

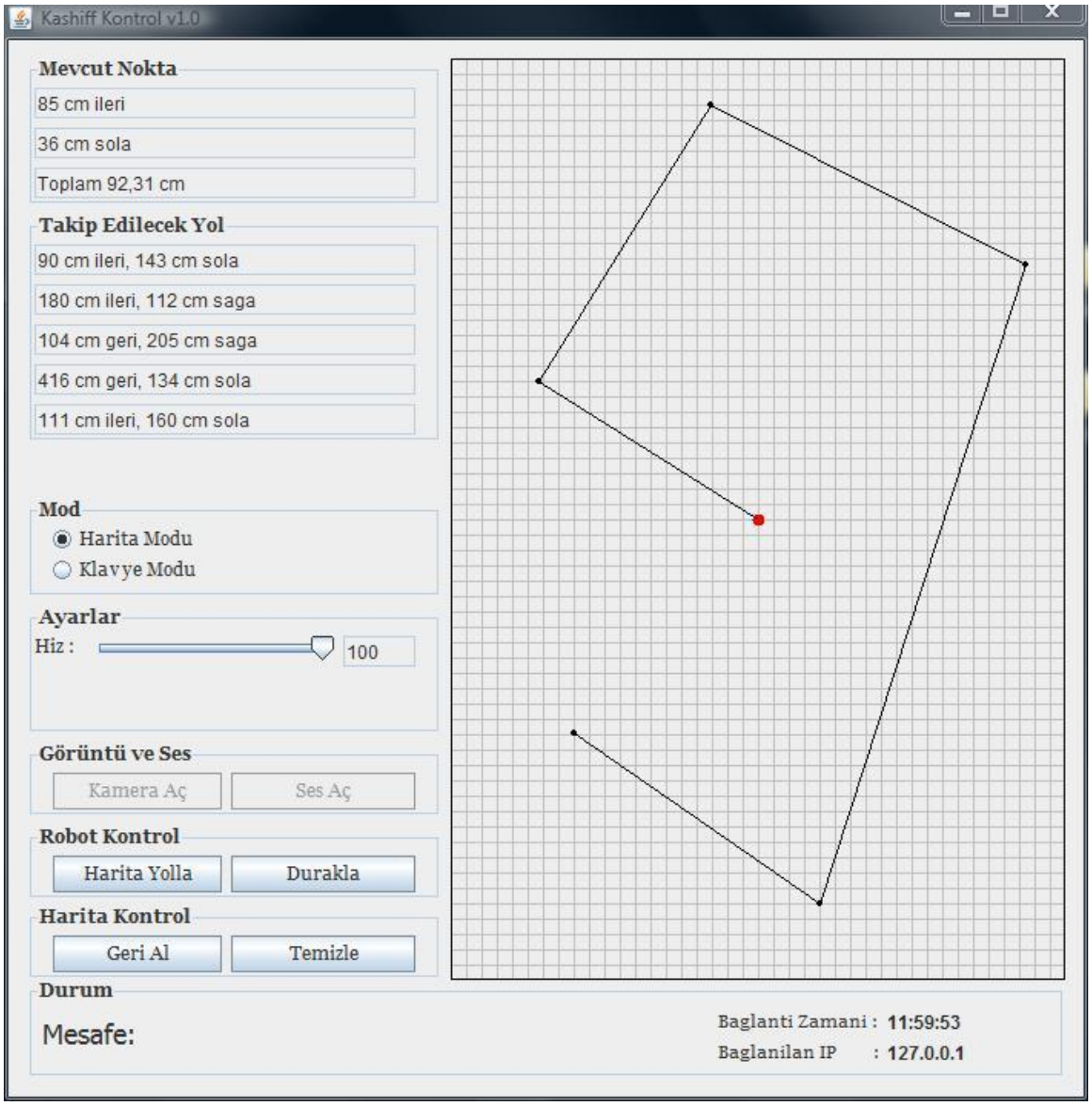
Kullanıcı kontrol programında robotun kontrolü iki farklı yöntemle sağlanabilmektedir. Bunlardan birincisi, kullanıcının sanal bir ekran üzerinde çizmiş olduğu yol üzerinde robotun hareket etmesini sağlayan harita modudur. Diğeri ise, kullanıcının bilgisayar klavyesinde bulunan yön tuşlarını kullanarak robotu hareket ettirmesini sağlayan klavye modudur.

6.2.1 Harita Modu

Robot kontrolü için kullanılabilecek modlardan bir tanesi “Harita Modu” dur ve bu mod kontrol programı çalıştırıldığı zaman seçili olarak gelen moddur.

Bu mod seçili olduğun zaman, kontrol programının sağ tarafında bulunan ızgaralı alan sanal harita olarak kullanılmaktadır. Sanal harita 4 metre en ve 6 metre boya sahip engelsiz ve düz bir alanı temsil etmektedir. Robotun bulunduğu yerin bu alanın tam ortasında bulunan kırmızı nokta olduğu ve yönünün de yukarı doğru olduğu kabul edilir.

Sanal harita üzerinde çizilecek olan sanal yol için en fazla beş varış noktası belirlenebilmektedir. Bu noktaların sayısını arttırmak teknik olarak mümkün olabilmekle beraber program yazılırken beş ile sınırlandırılmıştır. Şekil 6.8’de beş varış noktası da seçilmiş bir sanal yol örneği gösterilmektedir.



Şekil 6.8 Harita modu

Ekranın sol tarafında bulunan "Mevcut Nokta" alanında, fare sanal harita üzerinde gezdirilirken bulunulan noktanın en son işaretlenen varış noktasına göre konumu verilmektedir. Bu bilgi sayesinde, son işaretlenen varış noktası ile bu noktanın işaretlenmesi durumunda sağa-sola ve ileri-geri gidilecek mesafe bilgisi ile toplamda alınacak mesafe bilgisi gösterilmektedir.

Sol tarafta bulunan bilgi alanlarından bir diğeri ise "Takip Edilecek Yol" alanıdır. Bu alanda gösterilen bilgiler ise izlenecek olan sanal yol ile ilgili yazılı bilgilerdir. Bir önceki varış noktası ile işaretlenen varış noktası arasındaki sağa-sola ve ileri-geri gidilecek mesafe bilgilerini vermektedir.

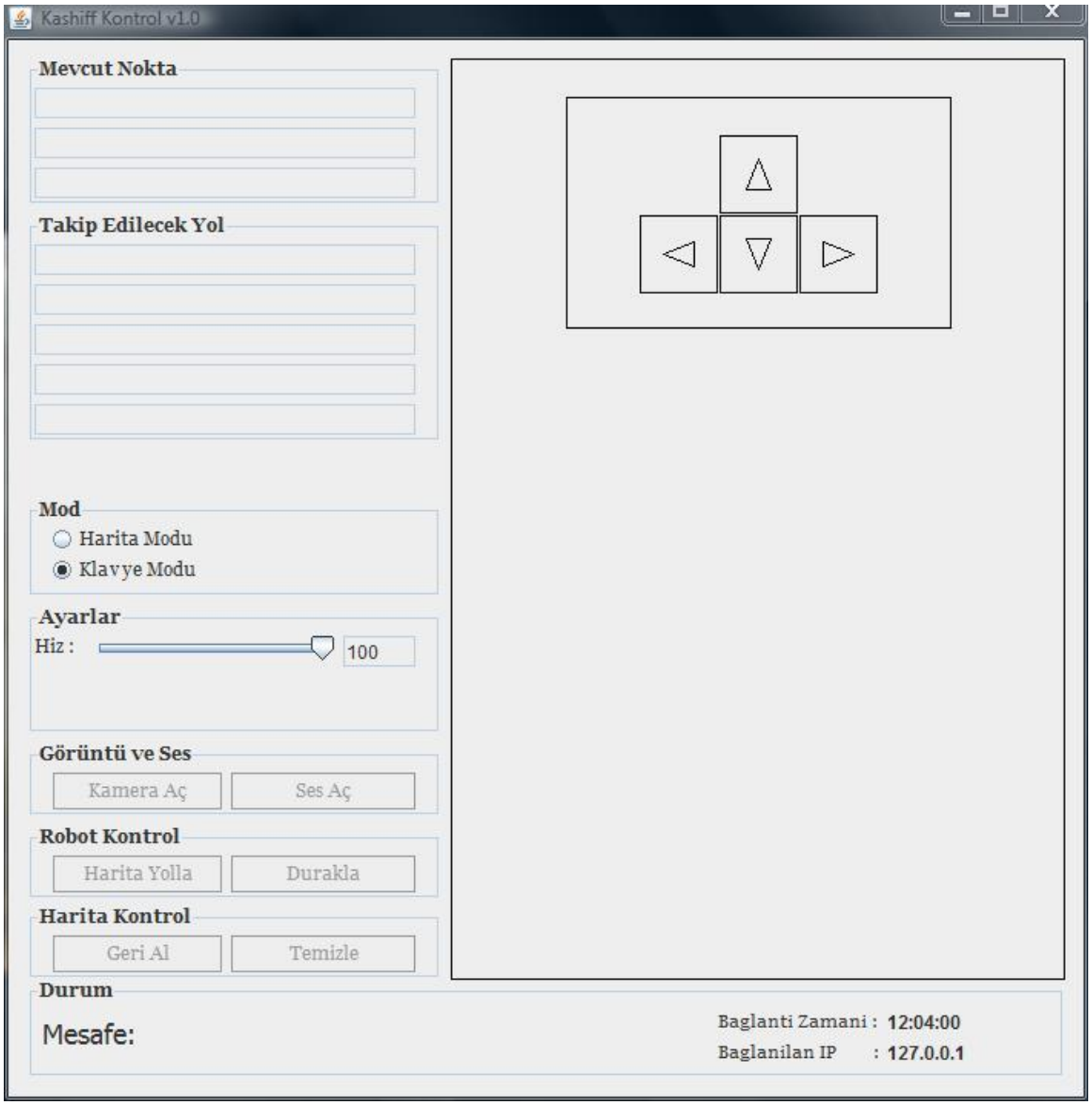
Sanal yol çizilirken yapılabilecek yanlış bir varış noktasını silmek için "Geri Al" butonu, sanal harita üzerindeki yolun tamamının silinmesi için ise "Temizle" butonu kullanılmaktadır. Sanal yol istenilen şekilde çizildikten sonra "Harita Yolla" butonuna basılarak robotun bu yola uygun şekilde hareketine başlaması sağlanır. Robot, karşısına herhangi bir engel çıkmaması durumunda bu yolu tamamlayıncaya kadar ilerleyecektir. Robotun karşısına herhangi bir engel çıkması durumunda robot hareketine son verecek ve hareketine devam etmek için başka bir kontrol komutu bekleyecektir. Bu durumda yapılması gereken, sanal harita üzerindeki yolun silinip yeni bir sanal yol çizilmesi ve robotun bu yola uygun olarak ilerlemesinin sağlanmasıdır. Robotun karşısına engel çıktığı durumlarda "Klavye Modu" kullanılarak robotun engeli aşmasını sağlamak da kullanılması muhtemel yöntemlerden biridir. Robot hareket ederken herhangi bir nedenden dolayı hareketini duraklatması istenirse "Durakla" butonu kullanılmalıdır. "Durakla" butonuna basıldığında bu buton "Devam Et" butonuna dönüşmekte ve robot bir dahaki komuta kadar beklemektedir. Bu aşamada "Devam Et" butonuna basarak robotun yolun kalanını tamamlaması sağlanabileceği gibi yeni bir sanal yolu kullanarak ilerlemesi veya "Klavye Modu" kullanılarak manuel olarak kontrol edilmesi de sağlanabilir.

"Harita Modu" kullanılırken "Ayarlar" alanında bulunan bar sağa-sola hareket ettirilerek robotun hızı yüzde cinsinden istenilen değere getirilebilir.

Ayrıca "Kamera Aç" ve "Ses Aç" butonları kullanılarak kamera ve ses ekranlarının açılması sağlanabilir. Kontrol programı açıldığında kamera ve ses ekranları kapalı olarak gelmektedir.

6.2.2 Klavye Modu

Bu mod, hakkında çok fazla bilgi sahibi olunmayan veya çok fazla engel olan alanlarda kullanılmak üzere tasarlanmış bir moddur. Şekil 6.9'da "Klavye Modu" kullanılırken açılan ekran görüntüsü verilmektedir.



Şekil 6.9 Klavye modu

"Klavye Modu" kullanılırken kontrol tamamen kullanıcıya ait olmakla beraber "Harita Modu" kontrolünde olduğu gibi robotun önüne herhangi bir engel çıkması durumunda robotun ileri yönlü hareket etmesi mümkün olmayacaktır. "Klavye Modu" seçildiğinde sağda bulunan ızgaralı alan yerine klavyenin ileri-geri-sağ-sol tuşlarını temsil eden sanal bir klavye görüntüsü gelmektedir. Klavyenin hangi tuşuna basıldığı bu sanal klavye üzerindeki tuşların renklendirilmesi ile anlaşılabilmektedir. "Klavye Modu" kullanılırken sol tarafta bulunan "Mevcut Nokta" ve "Takip Edilecek Yol" alanları herhangi bir bilgi göstermemektedir. "Robot Kontrol" ve "Harita Kontrol" alanları da pasif durumda bulunmaktadır.

"Klavye Modu" kullanılırken de "Ayarlar" alanında bulunan bar sağa-sola hareket ettirilerek robotun hızı yüzde cinsinden istenilen değere getirilebilir.

7. SONUÇ

Proje çalışmasının bitiminde, IP tabanlı ve kablosuz olarak kontrol edilebilecek bir mobil robot tasarlanmıştır. Yapılan uygulama testleri sırasında karşılaşılan problemler, tasarlanan robotun birtakım geliştirmelere ihtiyaç duyduğu sonucunu ortaya çıkarmıştır. Buna göre, projede kullanılan Windows XP işletim sistemi lisanslı olduğundan, projeye ek bir maliyet getirmektedir. İşletim sisteminde yapılacak bir değişiklik ile Linux gibi açık kaynak kodlu bir işletim sistemine geçmek ve böylece maliyetlerde düşüş sağlamak mümkün olabilmektedir. Fakat Windows işletim sistemi Linux işletim sistemine nazaran daha yaygın olarak kullanıldığından, bilgisayar üzerine takılan ek parçaların sürücü yazılımlarına erişim ve bunların ilgili işletim sistemlerine yüklenmesi daha kolay olmaktadır. Ayrıca projede kullanılan her parçanın Linux işletim sistemine uygun sürücü yazılımı da bulunmamaktadır.

Robotun sahip olduğu en önemli özelliklerden biri olan Wi-Fi haberleşmesi anakart üzerinde dahili Wi-Fi birimi olmamasından dolayı, harici bir USB Wi-Fi dönüştürücü kullanılarak gerçekleştirilmiştir. Bu dönüştürücüler dahili Wi-Fi birimleri ile karşılaştırıldığında hem daha düşük performansa sahip olmakta hem de gereksiz yer kaplamaktadır. Anakartta yapılabilecek bir değişiklik yardımıyla hem haberleşme performansının arttırılması hem de yer tasarrufu sağlanması mümkün olabilecektir.

Yapılan çalışma özgün bir çalışma olup, mikrodenetleyiciler üzerinde koşan C programlama dili kullanılarak yazılmış program, robotun hareketi ve kontrol edilmesi için Java programlama dili kullanılarak yazılmış kullanıcı arayüzleri ve haberleşme programlarının tamamı tarafımızdan tasarlanarak geliştirilmiştir. Kullanıcı kontrol programı ile robot kontrol programları arasındaki bağlantı ve veri alışverişi işlemleri sorunsuz bir şekilde gerçekleştirilmiş ve kullanıcı kontrol programından gönderilen veriler yardımıyla robot donanımının istenilen şekilde hareket ettirilmesi sağlanmıştır.

Robot üzerinde bulunan USB Wi-Fi adaptör aracılığıyla robot, kablosuz ağ bulunan alanlarda otomatik olarak IP adresi almakta ve kullanıcı bu IP adresi aracılığıyla eğer robota ait bağlantı parolasına sahipse robota bağlanabilmektedir. Buna ek olarak, robot üzerindeki USB Wi-Fi adaptörü iki farklı modda çalışabildiğinden, herhangi bir ağ kurmaya gerek kalmadan da robot ile direkt olarak noktadan noktaya kablosuz haberleşmek mümkün olmaktadır. Sistemin bu esnek yapısı sayesinde, aynı ortamda olmadan, kablosuz ağ mesafesi içerisinde ya da internet üzerinden çok uzak mesafelerden robotun kontrolü mümkün olabilmektedir. Robot üzerinde bulunan kamera sayesinde robotun bulunduğu ortamın görüntüsü ve ortamdaki ses de

kablosuz ağ aracılığıyla kullanıcıya iletilebilmektedir.

Herhangi bir kullanıcının robot ile bağlantı kurabilmesi için robota bağlantı parolasını bilmesi gerekmektedir. Bu parola sayesinde robotun IP adresini bilen başka birinin ya da robotun bağlı olduğu ağdan robotun IP adresini elde eden kişilerin robota bağlanıp kontrol etmeleri engellenmiştir. Böyle bir güvenlik özelliğine sahip olmasından dolayı benzer projelerden farklılık göstermektedir.

Kontrol programları yazılırken benzer fonksiyonları gerçekleştirebilen ancak platform bağımsız olmayan programlama dilleri yerine Java tercih edilmiştir. Bu sayede robotun kontrolünün sadece bilgisayar ile değil, günümüzün popüler iletişim araçları olan java uyumlu cep telefonları ve PDA'ler ile de yapılabilmesine zemin hazırlanmıştır. Mobilitenin önem kazandığı günümüzde robot kontrolünün bu şekilde gerçekleştirilebiliyor olması da projenin önemli özelliklerindendir.

Sonuç olarak robotun, kablosuz ağ mesafesi içerinde, bilgisayar ya da java uyumlu mobil cihazlar ile uzaktan kontrolü ve bulunduğu ortamın görüntü ve sesini kullanıcıya iletmesi işlemleri gerçekleştirilmiştir. Bu sayede insanların giremediği ya da girmesinin tehlikeli olduğu alanlarda, robotun yanında olmadan, gözlem ve keşif yapılabilmekte ve ortam hakkında bilgi sahibi olunabilmektedir. Yine benzer şekilde robotun, bomba imha, nükleer atık taşıma gibi insan hayatı ve sağlığı açısından tehdit oluşturabilecek görevlerde kullanımı da mümkün olmaktadır. Kontrol programlarının tasarımı yapılırken oluşturulan platform bağımsız yapı sayesinde ise robotun, geliştirilmeye açık olması ve farklı donanım ve işletim sistemleri üzerinde, değişikliğe gerek olmadan çalıştırılabilmesi sağlanmıştır.

Ayrıca 23-26 Aralık 2009 tarihleri arasında ODTÜ Kültür ve Kongre Merkezi'nde gerçekleştirilen Elektrik, Elektronik, Bilgisayar, Biyomedikal Mühendisliği 13. Ulusal Kongresi'nde mobil keşif robotunun sunumu yapılmış ve "Kablosuz Ağ Tabanlı Gezgin Keşif Robotu: Kaşif" isimli bildiri yayımlanmıştır.

KAYNAKLAR

Aslan, K., (2002), A dan Z ye C Kılavuzu, Pusula Yayıncılık, İstanbul.

Çavuşoğlu, İ. ve Kırmızı, F., (2007), "Seri Port ile Haberleşebilen Uzaktan Kumandalı Kameralı Araç", Bitirme Tezi, Yıldız Teknik Üniversitesi Elektronik ve Haberleşme Mühendisliği.

Çayıroğlu, İ. ve Şimşir, M., (2008), "PIC ve Step Motorla Sürülen Bir Mobil Robotun Uzaktan Kamera ile Kontrolü", Erciyes Üniversitesi Fen Bilimleri Enstitüsü Dergisi, 24 (1-2) 1 - 16 (2008)

Çölkesen, R., (2001), Network TCP/IP Unix, Papatya Yayıncılık, İstanbul.

Çölkesen, R. ve Örencik, B., (2002), Bilgisayar Haberleşmesi ve Ağ Teknolojileri, Papatya Yayıncılık, İstanbul.

Murphy, R., (2000), Introduction to AI Robotics, MIT Press, London.

Önal, A. ve Çelik, T., (2005), "Adım Motorlu Bir Mobil Robot Tasarımının RF Aracılığıyla Bir Bilgisayar Tarafından Kontrolü", Bitirme Tezi, Yıldız Teknik Üniversitesi Elektronik ve Haberleşme Mühendisliği.

Pekgöz, N., (2005), JAVA, Pusula Yayıncılık, İstanbul.

Sır, M. ve Umar M., (2007), "RF Üzerinden Bilgisayar Kontrollü Forklift Robot", Bitirme Tezi, Yıldız Teknik Üniversitesi Elektrik Mühendisliği.

Ünlü, B., (2007), "İnternet Üzerinden Mobil Bir Robotun Kontrolü", Bitirme Tezi, Yıldız Teknik Üniversitesi Elektronik ve Haberleşme Mühendisliği.

Pouli R., (1981), Robot Manipulators Mathematic, Programming and Control.

Yalvaç, M., (2008), "Bilgisayar Aracılığı ile Kablosuz Kontrol Edilebilen Mobil Araştırma Robotu Projesi", Bitirme Tezi, Düzce Üniversitesi Elektronik ve Bilgisayar Eğitimi.

Yazıcı, A., Parlaktuna, O. ve Özkan, M., "Mobile Robot Applications in Agriculture", Proceedings of the Fifth GAP Engineering Congress, 26-28 Nisan 2006, Şanlıurfa, 497-503.

Yıldırımoğlu, M., (2000), TCP/IP, Pusula Yayıncılık, İstanbul.

Yıldız, N., (2004), "Araba Benzeri Bir Gezgin Robotun Donanımı ile Yazılımının Tasarlanması ve Gerçekleştirilmesi", Bitirme Tezi, Yıldız Teknik Üniversitesi Elektronik ve Haberleşme Mühendisliği.

Yılmaz, N., Sağıroğlu Ş. ve Bayrak M., (2006), "Genel Amaçlı Web Tabanlı Mobil Robot: SUNAR", Bitirme Tezi, Selçuk Üniversitesi Elektrik Elektronik Mühendisliği.

Yurttakal, O., (2007), "Ultrasonik Sensöre Sahip Gezgin Robot Uygulaması", Bitirme Tezi, Yıldız Teknik Üniversitesi Elektronik ve Haberleşme Mühendisliği.

EKLER

Ek 1 RS232C protokolü

RS-232 Standardı en ilk olarak 1962 yılında çıkmıştır ve onun üçüncü versiyonu 1969 yılında RS-232C olarak adlandırılmıştır. RS-232D standardı ise RS-232C üzerinde genişletme yapmak için 1987 yılında çıkmıştır. RS232D standardı aynı zamanda EIA-232-D olarak da bilinir.

RS-232C Electronic Industries Assosiation (EIA) tarafından daha bilgisayarın başlangıç zamanları sayılan yıllarda tasarlanmıştır. EIA'da çalışan mühendisler gelecek için nasıl bir şey geliştireceklerinden pek emin değillerdi. Böylece RS-232C standardının mümkün olduğu kadar esnek olması için çok sayıda farklı sinyal hatları sağlamışlardır. Fakat günümüzde bu sinyal bağlantılarının çoğu kullanılmaktadır.

RS-232C 25 pin DB-25 konnektörü kullanması yanında bazı seri arabirimler, daha küçük olan DB-9 konnektörünü de kullanır. Bu konnektör 9 pinlidir. 1984 yılında IBM'in bilgisayarı AT'yi takdim etmesinde 9 pinli D tipi DB-9 konnektörünü kullanmıştır. IBM'in standartları belirlemede bir üstünlüğü olduğundan dolayı o zamanlarda bazı üretici firmalar tarafından 9 pinli konnektörler desteklenmiştir. Böylece DB-25 ve DB-9 olmak üzere iki çeşit konnektör kullanılmaya başlanmıştır. 9 pinli konnektörün çıkışına sebep olarak 25 pinli konnektörde kullanılan uçların hepsinin kullanılmadığını gösterebiliriz.

IBM uyumlu bilgisayarda PC'nin arkasına birçok port yerleştirilebilmektedir. Bu portlar farklı boyutlarda olabilmektedir. RS-232 seri portu genellikle COM1, COM2, RS-232 veya seri olarak belirlenmiştir. Eğer port isimli olarak belirlenmemişse, bağlantı için kullanılacak olan doğru portun bulunması önem kazanır.

Seri haberleşme asenkron ve senkron olmak üzere iki yöntemle yapılır. Alıcı ve gönderici iletişimi doğru yapabilmeleri için aynı iletim yöntemini kullanmalıdır. Asenkron haberleşmede karakter bazında haberleşme yapılır ve veriler herhangi bir anda gönderilebilir. Senkron haberleşmede ise veri ve verinin etrafında veriden türetilmiş kontrol alanları, adres alanları içeren bir blok yapı söz konusudur. Transfer işlemi veri bloğu tamamlanana ya da alıcı ve verici arasındaki senkronizasyon kaybolana kadar devam eder. Karakter blokları bazında haberleşme yapılır.

Bu sebeple küçük veri paketlerini iletiminde veri alanı diğer alanlardan daha az bir yer tutar. Bu da birim zamanda aktarılan veri miktarının azalması demektir. Bu sebeple yakın mesafedeki cihazlarla karakter tabanlı haberleşme yapılacağı zaman asenkron seri iletişim

kullanılır. Asenkron seri haberleşmede kullanılan veri yapısı "Başlama Biti", "Veri Bitleri", "Eşitlik Biti" ve "Durdurma Biti"ni içermektedir.

Gönderim işlemi başlangıç biti ile başlar. Başlangıç bitinden sonra gönderilecek karakter gelir. Her bir data biti için negatif gerilime lojik 1 karşılık gelirken, her bir dala biti için pozitif gerilimine lojik 0 karşılık gelir. Gönderilen bu karakter genellikle 7 bittir ve karakterin bir ASCII kodu vardır. Yani iletişim ASCII kodu ile yapılır. Bir veri gönderilirken, gönderilen veri LSB'den MSB'ye doğru akar. Karakterin gönderiminden sonra ise eşlik biti gönderilir. Bunun amacı yollanan veride tek bitlik hata olup olmadığının anlaşılmasıdır. Eğer tek (odd) eşlik seçilmiş ise bu bit yardımıyla yollanan birlerin sayısının tek olması sağlanır. Eğer çift (even) seçilmişse birlerin sayısı çift sayı yapılır. Veri paketinin sonunun belirtilmesi bir veya iki tane olan dur bitleri ile olur. Daha sonra yeni bir veri paketi için bu işlemler aynı sıra ile tekrarlanır. Karşı tarafta bulunan, başlangıç bitini tekrar aldığında bundan sonra bir karakterin gönderileceğini anlar.

Ek 2 Wi-Fi Teknolojisi

Kablosuz ağlar günümüzde bankacılıktan mesaj göndermeye, habercilikten televizyonculuğa kadar hemen her alanda çok yaygın bir şekilde kullanılmaktadırlar. Gelişen teknolojiyle birlikte yaşamın ayrılmaz bir parçası haline gelmiş, günlük hayatın olmazsa olmazları arasına girmişlerdir.

Kablosuz ağlar, basit bir yaklaşımla, iki ya daha fazla bilgisayarın birbirleriyle herhangi bir kablo kullanmadan, radyo frekansları aracılığıyla yönlendiriciler ve/veya aracılar kullanarak haberleşmesini sağlarlar.

Kablosuz ağların avantajları ve dezavantajları aşağıdaki gibi verilebilir.

Avantajları

- Bilgi transfer hızı küçük ölçekli yerel ağlar için yeterlidir (802.11b için 11 mbit/saniye).
- Hata giderici algoritmalar sayesinde güvenilir haberleşme sağlar.
- Şifreleme tekniği kullanarak haberleşmeler için gizlilik sağlanabilir.
- Açık alanda 300 metreye kapalı alanda ise 120 metreye kadar bilgi gönderilebilir.
- Kablosuz ağ erişim noktalarını mevcut ağa eklemek çok kolaydır.

Dezavantajları

- Kablosuz ağ donanımları geleneksel ağ donanımlarından daha pahalıdır.
- Kurulum işlemi geleneksel ağlardan daha zordur.
- Oda duvarları sinyal kalitesinin büyük miktarda düşmesine yol açar.

Kablosuz ağ kartı taşıyan bilgisayarların yeri değiştirildiğinde veri aktarımı hızında olumsuzluklar yaşanabilir.

Kablosuz haberleşmede iki alternatif mevcuttur. Bunlar birisi Ad hoc mod, diğeri ise infrastructure moddur.

Ad hoc ağ tamamen bağımsız olarak hareket eden bir grup kablosuz cihazdan oluşmaktadır. Bu cihazlara genel olarak düğüm adı verilir ve her düğüm bir yönlendirici (router) ve bir kullanıcıdan oluşuyormuş gibi düşünülebilir. Söz konusu düğümler herhangi bir merkezi

cihaz olmadan kendi aralarında haberleşebilmektedir. Haberleşme işlemi tamamlandıktan sonra, düğümler hareket etmeye devam edebilir.

En basit şekliyle, düğümler birbirlerinin kablosuz iletim alanlarında olduklarında aralarında direkt olarak iletişim kurabilirler.

Infrastructure mod ise, ortamdaki kablosuz ağ cihazlarının haberleşmesi için arada erişim noktası gibi bir cihaza ihtiyaç duyulmasıdır. Ad hoc moda göre biraz daha karmaşıktır ve özel olarak ayarlanmadıysa bilgisayar işletim sistemi bu modu kullanacak şekilde yapılandırılmıştır. Infrastructure modda kablosuz ağ istemcileri birbirleri ile direkt konuştuklarını düşünürler fakat tüm paketler erişim noktası aracılığı ile iletilir.

Burada ağa dahil olmayan herhangi bir kablosuz ağ cihazının tüm trafiği izleme riski vardır. Bu sebeple Infrastructure mod kullanırken genellikle iletişim şifrelenir. Şifreleme amaçlı olarak WEP ya da WPA gibi protokoller kullanılır. Şifreli iletişimde aradaki trafik izlense bile anlaşılmaz olacaktır.

Ek 3 Mikrodenetleyici Kaynak Kodları

Motor Kartı Mikrodenetleyici Yazılımı

```
#define cbi(port, bitnum) port &= ~(1 << bitnum)
#define sbi(port, bitnum) port |= (1 << bitnum)
#include <avr/io.h>
#include <stdbool.h>
#include <stdio.h>
#include <stdint.h>
#include <avr/interrupt.h>
#include <util/delay.h>

#ifndef F_CPU
#define F_CPU 4000000 // 4MHz
#endif

// UART
#define UART_BAUD_RATE 9600
#define UART_BAUD_CALC(UART_BAUD_RATE,F_OSC) ((F_CPU)/((UART_BAUD_RATE)*16L)-1)

#include <yeni.h>  // Fonksiyonlar
//SOL MOTOR,PORTB

int mod, bilgi[30], sayac, durakla, updated, aci[5], yon[5], mesafe[5], hiz,
sabit_hiz, vites_sayaci, min_hiz;
int ind=0, bypass=0;
int kullanim_modu=0; // 0 -> Harita | 1 -> Klavye
int gidis_yonu=0; // 0 -> Dur | 1 -> Ileri | 2 -> Geri | 3 -> Sola | 4 -> Saga
uint8_t uzunluk=0;

SIGNAL(SIG_USART0_RECV) { // USART RX interrupt
  uint8_t get;  // Seri haberlesme cebi
  get = UDR; // Seri porttan veriyi alıyoruz

  if (mod==1) //mod1, ozel karakter bekle
  {
    if (get == 'B'){
      mod =2;
    }
  } else if (mod == 2){//mod2, uzunluk bilgisini al
    uzunluk=get;
    sayac=0;
    mod=3;
  } else if (mod == 3){//mod3, bilgileri diziye yaz
    uzunluk=uzunluk-1;
    bilgi[sayac]=get;
    sayac++;
    if (uzunluk == 0){
      mod = 1;
      updated=1;
    }

  }

}


void update(void){

  updated=0;

  if (bilgi[0]=='M') // Motor datasý
  {
    if (bilgi[1]=='H') // Hareket et
    {
      durakla=0;
```

```
  ind=0;
  bypass=1;

  for (int i=0;i<5;i++)
  {
    int k = (5*i);
    yon[i]=bilgi[2+k];
    aci[i]=( (bilgi[3+k]*255) + bilgi[4+k] );
    mesafe[i]=( (bilgi[5+k]*255) + bilgi[6+k] );
  }

}
else if (bilgi[1]=='D') // Durakla
{
  durakla=1;
}
else if (bilgi[1]=='G') // Devam et
{
  durakla=0;
  vites_sifirla(99);
}
else if (bilgi[1]=='Z') // Hiz
{
  hiz = bilgi[2];
}
else if (bilgi[1]=='S') // Sorgulama
{
//transmit enable
}
else if (bilgi[1]=='K') // Klavye bilgisi
{
  if (bilgi[2]=='U') //Ileri
  {
    gidis_yonu=1;
    bypass=1;
  }
  else if (bilgi[2]=='D') //Geri
  {
    gidis_yonu=2;
    bypass=1;
  }
  else if (bilgi[2]=='L') //Sola
  {
    gidis_yonu=3;
    bypass=1;
  }
  else if (bilgi[2]=='R') //Saga
  {
    gidis_yonu=4;
    bypass=1;
  }
  else if (bilgi[2]=='S') //Dur
  {
    gidis_yonu=0;
    bypass=1;
  }

  kullanim_modu=1;
  durakla = 0;

}
else if (bilgi[1]=='X') // Harita moduna gec
{
  bypass=1;
  kullanim_modu=0;

  yon[0]='-';
    aci[0]=0;
    mesafe[0]=0;
```

```
      yon[1]='+';
        aci[1]=0;
        mesafe[1]=0;
      yon[2]='-';
        aci[2]=0;
        mesafe[2]=0;
      yon[3]='+';
        aci[3]=0;
        mesafe[3]=0;
      yon[4]='-';
        aci[4]=0;
        mesafe[4]=0;
    }
  }
}

void vites_sifirla(int gecikme){

  if (gecikme == 99){
    min_hiz=40;
    vites_sayaci=0;
    for(int i=0;i<20;i++)
      _delay_us(100);
  }
}

void pulse(void){

  sbi(PORTB,7);
  sbi(PORTB,3);
  for(int i=0;i<min_hiz;i++)
    _delay_us(10);

  cbi(PORTB,7);
  cbi(PORTB,3);
  for(int i=0;i<min_hiz;i++)
    _delay_us(10);

}

void hareket_et(int yon){

  if (yon =='+'){
    if (bit_is_set(PIND, 2)){
      sbi(PORTB,6);  //Yon ileri
      cbi(PORTB,2);  //Yon ileri
      mesafe[ind]--;
    }else{
      vites_sifirla(99);
      return; //Onunde engel var!!
    }
  } else {
    cbi(PORTB,6);  //Yon ileri
    sbi(PORTB,2);  //Yon ileri
  }

  pulse();

  vites_sayaci++;

  if ( (vites_sayaci == 5) & (min_hiz > hiz ) ){
    min_hiz--;
    vites_sayaci=0;
  }
}

void donus_yap(int yon){

  if (yon =='+'){
```

```
    cbi(PORTB,6);  //Yon
    cbi(PORTB,2);  //Yon
  } else {
    sbi(PORTB,6);  //Yon
    sbi(PORTB,2);  //Yon
  }

  pulse();

  vites_sayaci++;

  if ( (vites_sayaci == 15) & (min_hiz > sabit_hiz ) ){
    min_hiz--;
    vites_sayaci=0;
  }

}

void update_kontrol(void){

  if (updated==1){ // Update olduysa basa don
      update();
  }

}


int main (void)
{
  // Port ayarları
  DDRB = 0xFF; // Giris-cikis ayarlari
  DDRD = 0xFA; // Giris-cikis ayarlari

  sei();

  USART_Init(12); //Seri haberlesme icin baslangic ayarlari

  mod=1;
  hiz=7;
  min_hiz=40;
  sabit_hiz=60;
  vites_sayaci=0;
  updated=0;
  durakla=0;

  sbi(PORTD,3); // LED 2
  sbi(PORTD,4); // LED 3
  sbi(PORTD,5); // LED 4

  sbi(PORTB,6);  //Yon ileri
  sbi(PORTB,5);  //Reset 5V
  sbi(PORTB,4);  //Enable

  cbi(PORTB,2);  //Yon ileri
  sbi(PORTB,1);  //Reset 5V
  sbi(PORTB,0);  //Enable

  yon[0]='-';
    aci[0]=0;
    mesafe[0]=0;
  yon[1]='+';
    aci[1]=0;
    mesafe[1]=0;
  yon[2]='-';
    aci[2]=0;
    mesafe[2]=0;
  yon[3]='+';
    aci[3]=0;
    mesafe[3]=0;
```

```
yon[4]='-';
  aci[4]=0;
  mesafe[4]=0;

for (;;)
{
    int gecikme = 0;
  bypass=0;
  update_kontrol();

  if (kullanim_modu==0) // Harita modu
  {
    while ((aci[ind]>0) & (bypass==0)){

      donus_yap(yon[ind]);
      aci[ind]--;

      while (durakla ==1)  // Durakla datasi gelirse bekle
      {
        update_kontrol();
      }

      update_kontrol();

      gecikme = 99  ;
    }

    vites_sifirla(gecikme);
    gecikme=0;


    while ( (mesafe[ind]>0) & (bypass==0)){

      hareket_et('+');

      while (durakla ==1)  // Durakla datasi gelirse bekle
      {
        update_kontrol();
      }

      update_kontrol();
      gecikme = 99;
    }

    vites_sifirla(gecikme);
    gecikme=0;

    if (bypass == 0)
      ind++;

    if(ind==5)
      ind=0;

  } else { // Klavye modu

    while ((bypass==0) & (gidis_yonu!=0))
    {
      switch (gidis_yonu){
      case 1: hareket_et('+');   break;
      case 2: hareket_et('-');   break;
      case 3: donus_yap('-');   break;
      case 4: donus_yap('+');   break;
      }
      update_kontrol();
      gecikme = 99  ;
    }

    vites_sifirla(gecikme);
    gecikme=0;
```

```
    }
  }
return 1;
}
```

Ultrasonik Mesafe Ölçüm Mikrodenetleyici Kartı Yazılımı

```
//PORTB,5 trigger icin kullanıldı
//PORTD,6 capture icin kullanıldı

#define cbi(port, bitnum) port &= ~(1 << bitnum)
#define sbi(port, bitnum) port |= (1 << bitnum)
#include <avr/io.h>
#include <stdbool.h>
#include <stdio.h>
#include <stdint.h>
#include <avr/interrupt.h>
#include <util/delay.h>
#define SOLMOTOR PORTB
#define SAGMOTOR PORTC

// UART
#define UART_BAUD_RATE 9600
#define UART_BAUD_CALC(UART_BAUD_RATE,F_OSC) ((F_CPU)/((UART_BAUD_RATE)*16L)-1)

#include <ayar.h>

int mod, bilgi[30], updated, sayac;
uint8_t uzunluk=0;

//define times for start and end of signal
uint16_t rising, falling;

SIGNAL(SIG_USART0_RECV) { // USART RX interrupt

  uint8_t get;  // Seri haberlesmede cep olarak kullanıyoruz
  get = UDR; // Seri porttan veriyi aldık

}

//Timer1 capture interrupt service subroutine
ISR(TIMER1_CAPT_vect)
{
   /*This subroutine checks to see if itwas start of pulse (rising edge)
   or was it end (fallingedge)and performs required operations*/
```

```
   //if high level detected
  if ((bit_is_set(PIND, 6))){

    //save start time
    rising=ICR1;

    //set to trigger next  on falling edge
    TCCR1B = _BV(CS10);
  }else{

     //save falling time
     falling=ICR1;

     //rising edge triggers next
     TCCR1B = _BV(CS10)|_BV(ICES1);

     //variable to display values on LED
     int led;

     //calculate the pulse width
     led = falling - rising;

     //convert to centimeters (cm)
     led /=60;
     UDR = (uint8_t)led;

     if (led>50) {
             sbi(PORTD,2); sbi(PORTD,3); sbi(PORTD,4); cbi(PORTD,5);  }
     else if (led>20) {
             sbi(PORTD,2); sbi(PORTD,3); cbi(PORTD,4); sbi(PORTD,5);  }
     else {
             cbi(PORTD,2); cbi(PORTD,3); sbi(PORTD,4); sbi(PORTD,5);  }
   }
}


void pulse(void){

   //send pulse throught PORTB,0
   sbi(PORTB,0);

   _delay_us(30);

   //Set PORTB,0 to LOW and end pulse
   cbi(PORTB,0);
```

```
}

int main (void)
{

  DDRD=0x3F;
  DDRB=0xFF;
  USART_Init(12); //Seri haberleþme için baslangic ayarlari

  sbi(PORTD,2);
  sbi(PORTD,3);
  sbi(PORTD,4);
  sbi(PORTD,5);

  mod=1;
  updated=0;

   //Input Capture Interrupt Enable
   TIMSK = _BV(ICIE1);

   for (int i = 0; i < 10; i++) _delay_ms(10);

   //Enable Global interrupt
    sei();

   //Start timer counter and Enable input capture on rising edge
   TCCR1B = _BV(CS10)|_BV(ICES1);

   while(1){
    for(int i=0;i<2;i++)
      _delay_ms(100);
    ICR1=0;
    pulse();
    }

return 1;

}
```

ÖZGEÇMİŞ

Doğum tarihi	12.09.1982	
Doğum yeri	İstanbul	
Lise	1996-2000	Pertevniyal Lisesi
Lisans	2000-2004	Yıldız Teknik Üniversitesi Elektrik-Elektronik Fak. Elektronik ve Haberleşme Mühendisliği Bölümü

Çalıştığı kurum(lar)

2005-2008	Netaş Telekomünikasyon A.Ş.
2008-2010	Ericsson Telekomünikasyon A.Ş.
2010	Turkcell