DOKUZ EYLÜL ÜNİVERSİTESİ

FEN BİLİMLERİ ENSTİTÜSÜ

KAMERALI GÖRÜNTÜ ALANI İÇİNDE HAREKET KONTROLÜ

Onur KESKİN

Temmuz, 2009

İZMİR

KAMERALI GÖRÜNTÜ ALANI İÇİNDE HAREKET KONTROLÜ

Dokuz Eylül Üniversitesi Fen Bilimleri Enstitüsü

Yüksek Lisans Tezi

Mekatronik Mühendisliği Bölümü, Mekatronik Mühendisliği Anabilim Dalı

Onur KESKİN

Temmuz, 2009

İZMİR

YÜKSEK LİSANS TEZİ SINAV SONUÇ FORMU

ONUR KESKİN, tarafından **PROF. DR. EROL UYAR** yönetiminde hazırlanan **"KAMERALI GÖRÜNTÜ ALANI İÇİNDE HAREKET KONTROLÜ"** başlıklı tez tarafımızdan okunmuş, kapsamı ve niteliği açısından bir Yüksek Lisans tezi olarak kabul edilmiştir.

Prof. Dr. Erol UYAR

Danışman

Doç. Dr. Evren TOYGAR

Jüri Üyesi

Yrd. Doç. Dr. Zeki KIRAL

Jüri Üyesi

Prof.Dr. Cahit HELVACI

Müdür

Fen Bilimleri Enstitüsü

TEŞEKKÜR

Yüksek lisans tez çalışmamda bilgi ve yol göstericiliği ile danışman hocam Sayın Prof. Dr. Erol UYAR'a, Sayın Arş. Gör. Dr. Levent ÇETİN'e ve Sayın Arş. Gör. Özgün BAŞER'e, bilgileri ile projenin ortaya çıkmasında en az benim kadar çalışan değerli arkadaşım N. Engin TOKLU'ya, sonsuz destek ve güvenleri ile her zaman yanımda olan aileme teşekkürü borç bilirim.

Onur KESKİN

KAMERALI GÖRÜNTÜ ALANI İÇİNDE HAREKET KONTROLÜ

ÖZ

Bu tez çalışmasında, belirli bir yüksekliğe sabitlenmiş web kamerası yardımıyla alınan geri besleme bilgileri, kontrolcü bilgisayar tarafından yönetilen, bluetooth kablosuz iletişim protokolü ile bağlı "Lego Mindstorms NXT" ile hazırlanmış mobil araçların kontrolünde kullanılmıştır.

Mobil araçların tanımlamasında tek renkten oluşan üç parçalı renk etiketleri kullanılmıştır. Renk etiketleri kabarcık analizi yöntemi ile araçların konumlarını ve doğrultularını hesaplamakta kullanılmıştır.

Birden fazla mobil aracın birbirlerine ve tanımlanan engellere (gerçek ya da sanal) çarpmadan, kullanıcı tarafından verilen konumlara en uygun yoldan gitmek için A* yol bulma algoritması kullanılmıştır. Mobil araçların yollarının kesişmesi durumunda önceliği olan araca diğer araç yol vermektedir.

Proje kapsamında bulunan yazılım ve arayüzü Windows işletim sistemi ve .NET çerçevesi üzerine C# programlama dili kullanarak hazırlanmıştır.

Anahtar sözcükler: çok ajanlı sistemler, mobil araç, yol bulma, görüntü işleme, kabarcık analiz yöntemi

MOTION CONTROL IN CAMERA VISION SPACE

ABSTRACT

In this thesis study, mobile vehicles which are constructed from “Lego Mindstorms NXT” and connected via Bluetooth wireless communication protocol are managed by the controller computer. This computer receives feedback information with the help of the web camera which is fixed to a certain height.

Mobile vehicles have three-part single color labels to recognize. This color labels are used by blob analysis for finding each vehicles’ coordinate and orientation.

A* path finding algorithm is used for finding most suitable path towards the target coordinate without coming into collision with each other and defined obstacles (real or virtual).

Project software and its interface has been prepared by using C# programming language which is based on Windows operating system and .NET framework.

Keywords: multi agent systems, mobile vehicle, path finding, computer vision, blob analysis

İÇİNDEKİLER

BÖLÜM BİR
GİRİŞ

1.1 Amaç

Günümüzde birçok alanda kullanımı gittikçe yayılan robotların, zamanla kendi kendilerine daha çok işi başarmaları, diğer robotlarla iletişim halinde olup işbirliği yapmaları sağlanmalıdır. Robotların, kullanıcılarından aldıkları görevleri, kendi sahip oldukları yetenekler ve diğer robotlarla kurabildikleri iletişim becerileri yardımıyla başarmaları çalışılmaktadır. Bu çalışmalarda robotların dış dünyadan haberdar olmaları için bilgisayarlı görü yöntemleri de çok sık kullanılmaktadır. Bilgisayarlı görü yöntemleri ile sağlanan geri besleme bilgileri ile görüntü alanı içinde bulunan birden fazla mobil aracın kontrolü planlanmıştır.

1.2 Problem Tanımı

Projenin geçerliliğini kontrol etmek ve elde edilen bilgi ve verilerden sonuçlara ulaşmak için bir dizi problem senaryoları hazırlanmıştır. Bu senaryolar:

- Hedefe yönelik görev: Kullanıcı tarafından belirlenen hedef noktasına, diğer robotlara çarpmadan gidilmesidir.
- Sabit engelli görev: Basit hedefe yönelik görevin yetenekleri ile birlikte çalışma çevresinde bulunan sabit engellere çarpmadan istenen konuma en uygun yoldan ulaşılmasıdır.
- Hareketli engel görevi: Çalışma çevresindeki durumun değişmesi ile mobil aracın yolunun güncellenmesi ve istenen konuma gelmesidir.

1.3 Hedefler

Senaryolarda belirtilen örnek görevleri ve bu görevlerin kombinasyonu ile türetilebilecek diğer görevleri başarıyla yerine getirebilmesi, kullanıcılar ve diğer yazılımcıların kolayca uygulayıp, geliştirebilmesi de düşünülerek, “Çok Silahşorlar” adlı bir yazılım çerçevesi oluşturulmuş ve basit kullanıcı arayüzü kullanıma sunulmuştur.

BÖLÜM İKİ
TEORİK ARKA PLAN

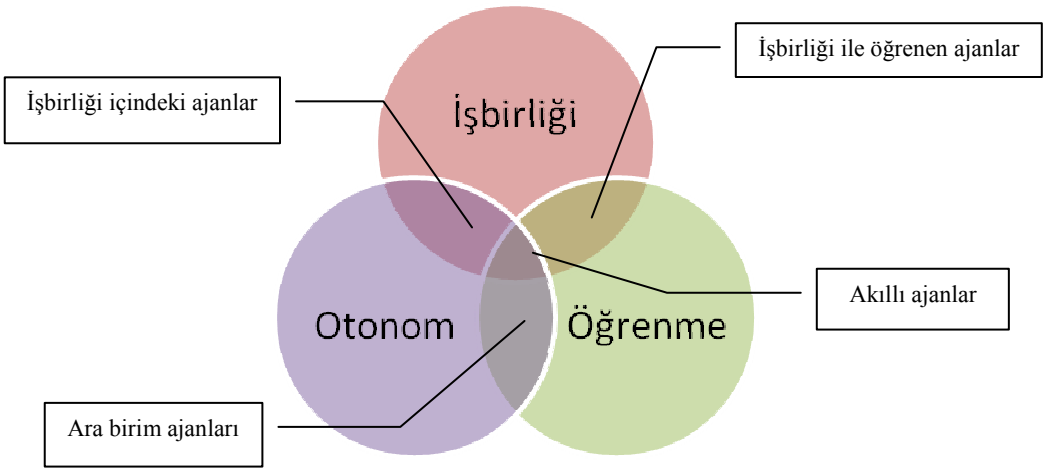
Proje, yapı olarak çok ajanlı sistemleri benimsemekte olup, ajan kavramı mobil araç ile gerçekleştirilmiştir. Mobil araç, çevresi ve diğer mobil araçlar hakkında sahip olduğu tüm bilgi bilgisayarlı görü yardımıyla temin edilmektedir. Elde edilen bilgiler ve verilen görevler doğrultusunda, mobil araçlar hareketlerini yol bulma algoritması yardımıyla hesaplar, engellere ve diğer mobil araçlara çarpmamak ve en uygun şekilde görevlerini tamamlayabilmek için yol planlaması yapmaktadırlar.

2.1 Çok Ajanlı Sistemler

Günümüzdeki yapay zekâ çalışmaları, "*akıllı ajan*" kavramına odaklanmıştır. Ajanın genel tanımı, algılayıcıları yardımıyla çevresi hakkında bilgi toplayarak, bu bilgiler doğrultusunda ve hareket mekanizmaları yardımıyla cevap veren bir yapıdır. Ajanın her zaman uygun performans ile cevap vermeyi denemesi, ajanın mantığının ölçütüdür. Örnek olarak; *insan ajan* (algılayıcı olarak gözleri, hareket mekanizması olarak elleri), *robot ajan* (algılayıcı olarak kamera, hareket mekanizması olarak tekerlekleri) ve *yazılım ajan* (algılayıcı ve hareket mekanizması olarak grafik kullanıcı arayüzü) gösterilebilir. Ajanlar kendi başlarına bir sistem oluşturabilecekleri gibi, birbirleri ile iletişim ve işbirliği içinde olarak çok ajanlı sistemleri de oluşturabilirler (Brachman,Dietterich,2007). Çok ajanlı sistemlerin kullanımının sağladığı faydalar aşağıda listelenmiştir:

- Asenkron ve paralel hesaplamalar yardımıyla hızlı ve verimlidir.
- Kullanılan ajanlardan birinin ya da birkaçının görevini başaramaması durumunda kalan ajanların yardımıyla sağlam ve güvenilirdir.
- Yeni ajanların sisteme kolayca eklenebilmesiyle ölçeklendirilebilir ve esnetilebilir.
- Gelişmiş tek bir sisteme göre ajanların maliyeti karşılaştırıldığında düşük maliyetlidir.

- Tek bir sisteme göre, ajanlardan oluşan modüler sistemin kolayca geliştirilebilir ve tekrar kullanılabilir.



Şekil 2.1 Ajanların yetenekleri (Anumba,Ugwu,Ren,2005)

Çok ajanlı sistemlerin mobil araç uygulamalarında (çok robotlu sistem) genellikle çevresinde hareket edebilen; yer, hava veya denizaltı araçları kullanılmaktadır. Ayrıca özel çalışma konularından olan, yeniden ayarlanabilen ya da modüler robotlar da kullanılmaktadır. Çok robotlu sistemlerin kapsamında, çoklu robot kol sistemleri ve iletişim ağı ile birleştirilmiş sistemler de bulunmaktadır. Çok robotlu sistemler hakkında yapılan başlıca araştırma konuları (Liu,Wu,2001):

- Çoklu robot yol planlaması
- Trafik kontrolü
- Düzen oluşumu, korunumu ve kontrolü
- Hedef takibi
- Kutu ittirme, Taşıma
- Yem arama
- Çok robotlu futbol
- Araştırma, Yerini belirleme
- Çarpışmadan kaçınma

Bu konular hakkındaki açıklamalar tablo 2.1'de yer almaktadır.

Tablo 2.1 Çok robotlu sistemlerin uygulama örnekleri

Uygulama örnekleri	Açıklaması
Yem arama ve gözetleme	Genellikle sürü robotlarının denenmesinde kullanılan çalışmalardır. Biyolojik sistemlerde gözlenen davranışlar temel alınır.
Toplanma ve biçimlenme	Robotların, birbirlerinin konumlarına göre hareketlerini düzenlemeleri ve belirli bir düzende toplanmaları ya da düzeni koruyarak hareket etmeleri sağlanır.
Kutu ittirme ve işbirliği içinde işleme	Çok robotlu işbirliği çalışmalarının temelini oluşturur. Tek robotun üstesinden gelemeyeceği durumlarda robotların işbirliği yapması sağlanır.
Çoklu hedef gözlemi	Çoklu robotla, çevrede bulunan birden fazla hedefi gözlemek ve incelemeye çalışırlar. Robot sayısından fazla hedef sayısı olması durumunda, robotların koordinasyonu çok önemlidir.
Trafik kontrolü ve çok robotlu yol planlama	Robotlar aynı çevrede çalışırken, hareketlerini diğer robotlarla çarpışmamak için düzenlemelidirler.
Futbol	Koordinasyon ve kontrol çalışmaları için temel oluşturur.

Çok robotlu sistemlerin ilk çalışmaları 1980'li yıllarda olmuş, günümüze kadar önemli ölçüde büyüyen bir araştırma konusu haline gelmiştir. En genel seviyede, çok robotlu çalışmalar iki temel kategoriye sahiptir; "*toplu sürü sistemleri*" ve "*özellikle işbirliği*" içinde bulunan sistemler. Toplu sürü sistemlerindeki robotlar kendi görevlerini, sürüde bulunan diğer robotlar hakkında aldıkları en az bilgi ile tamamlamayı amaçlarlar. Bu çeşit sistemler genellikle çok sayıda türdeş robotlardan oluşur. Diğer yandan özellikle işbirliği içinde bulunan sistemlerde robotlar çalışma çevrelerindeki diğer robotların mevcut durumlarından haberdar olup, takım arkadaşlarının durumlarına, hareketlerine ve ya yeteneklerine uygun biçimde aynı amaca yönelik harekette bulunurlar. Özellikle işbirliği içinde bulunan sistemler, "*güçlü işbirliği*" ve ya "*zayıf işbirliği*" adı ile iki ayrı biçimde değerlendirilirler. Güçlü işbirliği çözümünde, robotların hareketi sırası önemli olmayan aynı amaçlara

yöneliktir. Bu yaklaşımda, robotların kendi aralarında iletişime ve eş zamanlı çalışmaya ihtiyaçları vardır. Zayıf işbirliği çözümlerinde, robotların belli aralıklarla kendi başlarına bağımsız biçimde çalışmalarına izin verilmektedir. Özellikle işbirliği içinde bulunan çok robotlu sistemlerin robot üyeleri kendilerine ait algılayıcı ve etkileyici yeteneklerine sahip ayrışık robotlardan oluşabilmektedir. Bu durumdaki robot toplulukları, birbirlerinin yerlerine kullanılamayacakları için sürü sistemlerinden çok farklı bir kontrole sahiptirler.

Çoklu mobil araçların işbirliğinin kullanıldığı ana çalışma konuları Tablo 2.2 de olduğu gibi belirli başlıklar altında toplanabilir.

Tablo 2.2 Çok robotlu sistemlerin çalışma konuları

Çalışma konusu	**Açıklaması**
Mimari	Tüm çok robotlu sistemlerle ilgili olup, robot grup üyelerinin nasıl organize olduklarını ve iletişim kurduklarını inceler.
İletişim	
Sürü robotları	Çok robotlu sistemlerin belirli bir türüdür ve çok sayıda türdeş robotlardan oluşurlar. Birbirlerine güçlü bir şekilde bağlıdırlar.
Ayrışıklık	Yetenekleri açısından farklı robotlardan oluşan sistemlerdir.
Görev tayini	Ayrışık robotlardan oluşan sistemlerde, yeteneklerine uygun işlere karar veren mekanizmadır.
Öğrenme	Çok robotlu gruplara belirli konularda zaman içinde yeni davranışlar kazandıran mekanizmadır.

2.1.1 Mimari

Çok Robotlu Sistemlerin kontrol mimarisinin tasarımı, sistemin sağlamlığına ve ölçeklendirilebilmesine doğrudan etkisi bulunur. Çok robotlu gruplar, robot mimarisi olarak, tek robotlu sistemlerle benzer temellere sahip olsalar da, robotlar arası iletişim ve grup davranış özellikleriyle ayrılırlar. Çok robotlu gruplarda; merkezi, aşamalı, dağıtılmış ve melez olmak üzere yaygın olarak kullanılan mimariler bulunmaktadır (Siciliano,Khatib,2008).

Merkezi mimaride, tüm grup teorik olarak tek bir noktadan kontrol edilir. Bu sayede her grup üyesi, tüm grup hakkında net bir bilgiye sahip olur ve kontrol değişkenleri tüm gruba rahatlıkla iletilebilir. Dezavantajı, sayıca büyük grupların takip edilmesi ve eş zamanlı kontrolü zor ve maliyetli olabilir.

Aşamalı mimaride, her robot kendi çevresini gözler ve kendine verilen görevleri yapmakla sorumludur. Görevin tamamlanması her aşamadaki robotların kendine ait görevleri başarmasından geçer. Robotların kontrolünde askeri bir emir-komuta zincir yapısı mevcuttur. Pratikte uygulanması daha olası olsa da, dezavantajı üst kademedeki robotların hata toleranslarının düşük ya da hiç olmamasıdır. Bu mimariye örnek olarak ALLIANCE yapısını gösterebiliriz (Siciliano,Khatib,2008). Bu yapı, düşük seviyeli davranışların bulunduğu davranış kümelerinden ve birbirleri ile etkileşimi olmayan isteklendirme parametrelerinden oluşur. İsteklendirmeyi, robotun davranış kümesiyle başarabileceği görevlere karşı oluşturulan önceliği destekler. Bu yaklaşımda isteklendirme en başta sıfırdır. Her bir adımda, aşağıdaki maddelere bağlı olarak tekrardan hesaplanır:

- Önceki isteklendirme seviyesi
- Öncelik değeri
- Algılayıcılardan gelen geri bildirimin davranış kümesini desteklemesi
- Başka bir robotun davranış kümesinin etkileştirilmesi
- Başka bir robotun bu görev üzerinde çalışmaya başlaması
- Çalışma süresi gözlenerek, görev üzerinde çalışan robotun pes etmesi

Bu isteklendirme yukarıdaki her adımda artacak olsa da, aşağıdaki durumlar bunu azaltma yönünde olacaktır:

- Algılayıcılardan gelen geri bildirime göre artık görevin yapılmasına gerek kalmaması
- Robotun farklı bir davranış kümesinin etkinleşmesi
- Başka bir robotun ilk görev olarak, mevcut görevi alması
- Robotun görevi kabul etmesi

Dağıtılmış mimaride, grup içindeki robotlar tamamen bireysel olarak, sahip oldukları bilgiler yardımıyla verilen görevleri yapar. Bu mimari, robotların birbirine herhangi bir bağları bulunmaması sebebiyle diğer kontrol mimarilerine oranla çok daha sağlamdır. Dezavantajı, genel bir tutarlılık gerektiren görevlerde birlikte uyum içinde çalıştırılması zordur. Bu mimarinin ilk uygulamalarından olan "Mataric" 'in "Nerd Herd" sürü robot topluluğu, 20 adet türdeş robottan oluşmaktaydı (Siciliano,Khatib,2008). Robotlar engellerden kaçınma, hedef arama, kümelenme, yayılma, takip etme ve güvenli mesafeyi koruyarak dolaşma özelliklerini yerine getirebilmekteydi. Örneğin kümelenme için kullanılan mantık kod taslağı 1.1'de verilmiştir.

```
Eğer robot kümelenme mesafesi dışında ise;
      kümelenme merkezine doğru yönel ve git.
Değilse;
      dur.
```

Kod taslağı 1.1 Kümelenme mantığı

Bu çalışma sayesinde, düşük seviyeli davranışların birleşimi yardımıyla ortaklaşa davranışlar türetilebildiği gözlenmiştir. Örneğin sürü halinde dolaşılması için güvenli mesafeyi koruyarak dolaşma, kümelenme ve yayılma davranışları birleştirilmiştir.

Melez mimaride, yerel kontrol ile yüksek seviyeli kontrol yaklaşımını, sağlamlık ve tüm grup ile uyumlu çalışılabilmesi için birleştirilmiştir. Bu uyumluluk, amaçları, planları ve kontrol parametrelerini kapsar. Birçok uygulamada bu mimari uygulanmaya çalışılmaktadır.

2.1.2 İletişim

Çok robotlu sistem araştırmalarında, robotların birbirlerinin durumundan ya da eylemlerinden haberdar olması temel varsayımdır. Bu sayede genel bir uyum sağlanmış olunur. Bu bilgiler birkaç yöntemle elde edilebilir:

- Çalışma alanı içinde örtülü iletişimin sağlanması; robotlar diğer robotların eylemlerinin sonuçlarını algılayarak kendi eylemlerini gerçekleştirebilir.
- Gruptaki diğer robotların hareketlerini doğrudan gözlemeyerek, pasif eylem tanımlaması yapılabilir.
- Robotlar birbirleri ile kesin olarak iletişim halindedirler, bunu kablosuz ya da kablolu iletişim yöntemleri ile sağlayabilirler.

Tüm bu iletişim yollarının birbirlerine karşı üstünlükleri ve zayıflıkları vardır. Örtülü iletişim basit ve bağımsız olmasına karşın, kesin iletişim de bilginin aktarılması için iletişim kanallarına ve protokollere ihtiyaç duyulur. Bu kanallarda oluşabilecek gürültü ve bant genişliği sınırları düşük seviyedeki hatalara toleranslı olmasını sağlamaktadır ve güvenilirliğine zarar vermektedir. Örtülü iletişimde robotların algıladıkları durumlar limitli olabilir ya da yanlış yorumlanabilir. Kesin iletişimde doğrudan iletişiminin avantajı tüm grup üyelerine kesin olarak amaç ve/veya eylem durumunun bildirilmesidir. Bu sayede eylemlerde uyumluluk, birbirleriyle anlaşmaları ve bilgi transferi sağlanmış olur. Pasif eylem tanımlaması da herhangi bir bant genişliğine ve hata yapan bir iletişim mekanizmasına ihtiyaç duymaz.

Uygun iletişim tipinin seçilmesinde, çok robotlu sistemlere verilecek görevlerle doğrudan ilişkisi vardır. İletişimin tipine göre sağlanacak yararlar, maliyetler, güvenilirlik seviyeleri ve çalışma performansları dikkate alınması gereken maddelerdir.

İletişim konusuyla ilgili bir diğer çalışmada, hareket halindeki robotların bağlantılarını dinamik olarak sağlamasıdır. Bu sayede iletişimin kopması durumunda robot, kendiliğinden bunu onarabilecektir. Bunun başarılması için robotun, iletişim

ağı içinde kalacak şekilde eylemlerini düzenlemesi gerekmektedir. Bu sayede dinamik iletişim ağı kurulabilir.

2.1.3 Ayrışıklık

Robotların ayrışıklığı, davranışların, biçimlerinin, performanslarının, boyutlarının ve öğrenme kapasitelerinin farklılığı olarak tanımlanır. Genellikle büyük ölçekli çok robotlu sistemlerde türdeş robotların paralelliğinden ve tamamen birbirlerinin yerine geçebilir, yedeklenebilir olmalarından yararlanılır. Buna rağmen farklı tipteki robotların eş zamanlı kullanılmasını gerektiren görevlerde olabilir. Bu durumda tek bir robota tüm algılayıcıları ve etkileyicileri vermek yerine, ayrışık çok robotlu sistemleri kullanırız. Örneğin farklı boyutlardaki robotlar ile büyüklerin ulaşamadığı yerlere daha ufak olan robotlar ulaşabilir ya da maliyeti fazla algılayıcıları tüm robotlarda kullanmak yerine gerektiği yere sadece o algılayıcıya sahip grup üyesi robot yönlendirilebilir.

Ayrışıklık bazı durumlarda ilerideki görevleri düşünülerek uygulanabileceği gibi bazı durumlarda da uygulanması zorunludur. Maliyet düzenlemesinin yapılabileceği gibi ihtiyaçları tam olarak sağlayan robot grupları tasarlanarak, mühendisliğin gerektirdiği düşünce yapısını da ön plana çıkarmış olur. Ayrışıklığın zorunlu olduğu durumlara örnek olarak, grubun her ne kadar türdeş olması istenilse de teknik olarak üstesinden gelinemeyecek durumlarda, kullanılan türdeş algılayıcıların menzilinde bulunan farklılık gibi, grubun yeteneklerinin farkında olunması önem taşımaktadır.

2.1.4 Görev Dağılımı

Birçok çok robotlu sistemde, grubun amacı tanımlanan görevleri tamamlamaktır. Her görev çeşitli robotlar tarafından yapılabileceği gibi, her robot çeşitli görevleri de yerine getirebilir. Birçok uygulamada, görev bağımsız ya da aşamalı birkaç alt görev oluşturur. Bir kez görev kümesi ya da alt görevler belirlendiğinde, yapılması gereken görevlerin robotlara eşlenmesidir.

Görev dağılımını, tek robot görevleri ve çok robot görevleri olarak iki prensibe sınıflandırabiliriz. Tek robot görevlerinde, belirli bir zamanda görevin yapılması için tek bir robot yeterli olurken, çok robotlu görevlerde, aynı zaman zarfında aynı görevi birden çok robotun yapması gerekmektedir. Genellikle tek robotlu görevlerde, görevlerin bağımlılığı en az seviyede olduğu için bunlar zayıf işbirliği çözümleri ve gevşek bağlı görevler olarak adlandırılır. Çok robot görevlerinde birbirlerine bağlı alt görevlerden oluştuğu için güçlü işbirliği çözümleri ve sıkı bağlı görevler olarak adlandırılır. Bu durumda yüksek seviyede uyumlu çalışma ya da işbirliği gerektirir. Her görevin, alt görevlerin mevcut durumundan küçük bir zaman gecikmesiyle haberdar olmaları gerekir.

Robotlar ayrıca, belirli bir zamanda tek bir görev üzerinde çalışmaları durumunda tek görevli ve belirli bir zamanda birden fazla görevi işleyebilme durumlarında ise çok görevli robotlar olarak da sınıflandırılabilir.

Görev dağılımlarına göre bir sınıflandırma yaklaşımı da örnekleme çeşitlerine göredir. Kabaca davranış tabanlı görev dağılımı ve market tabanlı (görüşme tarzı) görev dağılımı olarak iki yaklaşıma ayrılabilir. Davranış tabalı görev dağılımında, genellikle robotlar görev dağılımlarını, kendi görevlerini açıkça tartışmadan belirlemeleri sağlanmıştır. Bu yaklaşımda, robotlar, grubun amacındaki mevcut durum bilgisine, grup üyelerinin yeteneklerine ve hangi robotun hangi görevi uygulayacağı bilgisine sahibidirler. Market tabanlı (görüşme tarzı) görev dağılımında robotlar arası iletişim esastır. Yapılması gereken göreve, robotlar, yeteneklerine ve uygun olma durumlarına göre girişimde bulunurlar. Robotların görev dağılımında market teorisinden yararlanılır ve belirli bir görev için en uygun robotun seçilmesi sağlanır.

2.1.5 Öğrenme

Çok robotlu öğrenme, yeni işlevsel davranışları ya da diğer robotların durumlarını öğrenmeyi amaçlar. Diğer robotların, çalışma çevresinde, her ne kadar kendi amaçları olsa da, bununla birlikte paralel olarak öğrenebilirler. Çok robotlu öğrenme,

tek robotlu öğrenmenin yanında sürekli durum ve hareket uzayını kapsarken; sınırlı eğitim süresini, yetersiz eğitim bilgisini, algıda ve paylaşılan bilgide oluşan belirsizliği problemleriyle de ilgilenmelidir.

Çok robotlu öğrenmede çalışılan konuların başında; çoklu hedef gözlemi, hava filosu kontrolü, av-avcı ilişkisi, kutu ittirme, yiyecek arama ve çok robotlu futbol gelmektedir.

2.1.6 Karar Verme

Bir grup ajanın, birlikte bulunduklar çevre içinde eşzamanlı verdikleri kararları incelemek için oyun teorisinden faydalanabiliriz. Stratejik oyun, bir başa deyişle normal kapsamlı oyun, ajanların etkileşiminin ne basit oyun teori modelidir (Brachman,Dietterich,2007).

Oyun teorisi, birbirleriyle belirsizlik durumu altında etkileşim halinde olan ajanların davranışlarını iki önermeye dayanarak anlamaya çalışır. Bu önermelerden biri, etkileşimde olan ajanların mantıklı olmaları, diğeri ise ajanın, diğer ajanların kararlarını dikkate alarak stratejik düşünebilmesidir. Ajanların, eylemlerini seçmelerine göre oyun farklılaşmaktadır. Stratejik oyunda, her ajan kendi stratejisini tek bir kez, oyun başlamadan önce seçer ve daha sonra tüm ajanlar eylemlerini eş zamanlı olarak uygularlar. Kapsamı geniş oyun ise, ajanlar, oyun içinde, planlarını tekrar düşünebilir ve tam doğru olmasa da diğer ajanların eylemlerinden haberdar olabilirler.

Özetlersek, stratejik oyunda her ajan tek bir eylem seçer ve seçilmiş olan ortak eyleme bağlı sonucunu alır. Bu ortak eylem, oyunun sonucu olarak tanımlanır. Oyundaki çözüm, oyunun sonucunu, tüm ajanların mantıklı oldukları ve stratejik düşündükleri varsayımı yardımıyla tahmin etmektir. İki ajanlı özel durumlarda, stratejik oyun, sonuç matrisi yardımıyla grafiksel olarak da gösterilebilir (bkz. Tablo 2.3). Bunun en bilindik örneği, tutsak ikilemidir. İki şüpheli ayrı ayrı, hiçbir iletişimleri olmayacak şekilde sorguya alınıyor. Şüphelilere bir anlaşma sunuluyor;

eğer biri suçu itiraf eder, diğeri sessiz kalırsa, itirafçı serbest, sessiz kalan ise 5 yıl cezaya çarptırılıyor. Eğer ikisi de itiraf ederse 3 yıl, ikisi de sessiz kalırsa hiçbir cezaya çarptırılmıyorlar. Şüpheliler sessiz kalıp, 5 yıl ceza çekmek yerine, itiraf edip ya 3 yıl ceza çekmeyi ya da hiçbir ceza almamayı tercih ediyorlar. Çünkü her ikisi de, en fazla kazanç elde etmeyi isterler.

Tablo 2.3 Tutsak ikilemi sonuç matrisi

	İtiraf etmek	Sessiz kalmak
İtiraf etmek	3,3	0,5
Sessiz kalmak	5,0	0,0

Sıkı rekabet ya da sıfır toplam tipine uygun, eşleşen kuruşlar örneğini verilebilir. A kişisi, B kişisine, yazı-tura oyununda farklı sonuçlar gelirse atılan kuruşu ödeyecek, aynı sonuçlar gelme durumunda ise B kişisi, A kişisine atılan kuruşu ödeyecektir (bkz. Tablo 2.4). Bu oyunda, her koşul $u_A(a) + u_B(a) = 0$ sonucuna götürür.

Tablo 2.4 Sıkı rekabet (sıfır toplam) sonuç matrisi

	Yazı	Tura
Yazı	1,-1	-1,1
Tura	-1,1	1,-1

Benzer durum, koordinasyon durumu içinde geçerlidir. Örneğin iki araç sürücüsü, aynı kavşaktan en önce kendisinin geçmesini istiyor fakat ikisi de aynı anda geçmeleri durumunda kaza yapacaklarının farkındalar, bu durumun sonuç matrisi tablo 2.5'deki gibi olur.

Tablo 2.5 Kavşak geçişi probleminin sonuç matrisi

	Geç	Dur
Geç	-1,-1	1,0
Dur	0,1	0,0

2.2 Yol Bulma ve Yol Bulma Algoritmaları

Hedefe yönelme problemi, mobil robotiğin klasik problemlerindendir. Mobil aracı yöneltmek için; kendi konumunu, diğer cisimlerin (engel, ara hedef noktaları, diğer araçlar) konumlarını ve bulunduğu noktadan diğer noktalara nasıl gidebileceğini bilmesi gerekir. Başlangıç noktasının bilinmesine bazen gerek olmasa da, en önemli soru robotun hedefe nasıl ulaşacağıdır. Yönlendirme ve mobil aracın hareket kontrolünde yolun bulunmasının yanında, en uygun yolun bulunması da gerekmektedir. En uygun yolun değerlendirmesi durumlar arasında farklılık gösterebilir (Kordic,Lazinica,Merdan,2005). Örneğin alınan yol miktarı, harcanan enerji miktarı, tehlikeli durumlara maruz kalma süresi gibi.

Yol bulma, kullanılacak yolun çevre temel alınarak seçimi ve tanımlanmasıdır. Yol planlama, seçilen teorik yol planının robotun fiziksel özelliklerine göre çıkarılmasıdır. Yol bulma algoritmalarının amacı, her hangi iki nokta arasındaki en iyi yolu, engelleri aşarak bulmaktır (Bourg,Seeman,2004). Bu iki tanım kavramsal olarak birbirlerinden farklı olsa da, birlikte çalışmalıdırlar. Yol bulma işlemi planlanan yolları değerlendirerek gerçekleşir. Ve yol planlama sistemine geri besleme yollayarak gidilmesi imkânsız seçeneklerin tekrar düşünülmesini sağlar. Yol planlama algoritmaları iki anahtar özelliğe sahiptir, eksiksizlik ve uygunluk. Çalıştığında, eğer çarpışmaların olmadığı bir yol buluyorsa ya da hiçbir yol bulamıyorsa, eksiksizdir. Başlangıçtan, hedef pozisyonuna en kısa ya da en iyi yolu buluyorsa, uygundur.

Tek araçlı yol planlama, A noktasından B noktasına belli kısıtlamalarla birlikte riskleri en aza indiren ya da risklerden kaçınan bir yol bulmak ve adımları planlamaktır. Genelde yol planlama için dört farklı yöntem kullanılır (Keskin,2008):

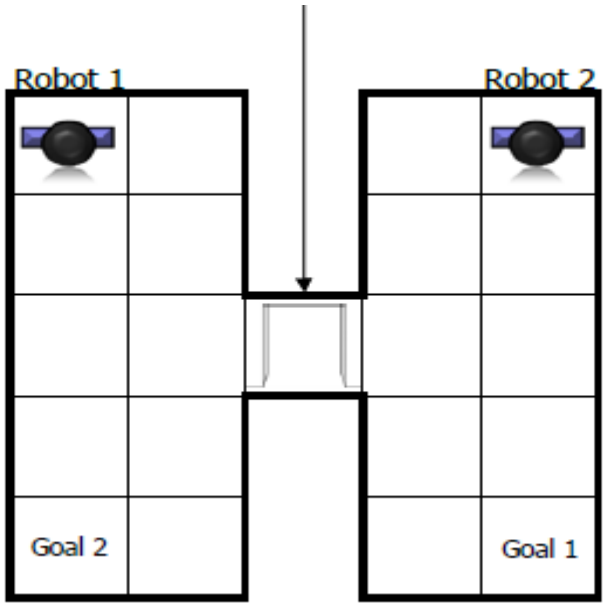
- *Yörünge iyileştirme*: ölçülebilen değerleri değiştirerek verilen sınırlar içinde en az maliyet getiren yörüngeyi bulmayı amaçlar. Çözümü en uygun kontrol problemlerinin çözüm yöntemleriyle benzerlik gösterir.
- *Rota planlama*: genelde işlemsel anlamda yörünge iyileştirme yerine tercih edilir. Rota planı yapılırken ara noktaların kümesi içinden hedefe ulaşmak için

geçilecek noktalar uygun biçimde sıralanır. Bu sıralama fazla ara noktalı kümelerde her zaman en uygun yolu bulmayabilir.

- *Benzeşim tabanlı yol planlama*: var olan planlama problemini daha önceden çözülmüş fizik, geometri ya da biyoloji içerikli problemlere benzeterek uygun bir yol planı oluşturmayı hedefler.
- *Oyun teorili yol planlama*: mantığında genelde bir av-avcı ilişkisidir, avcı avı ile arasındaki mesafeyi mümkün oldukça yakın tutmaya çalışır.

Çok araçlı yol planlama, tek araçlı yol planlamada kullanılan yöntemlerin birini ya da birkaçını kullanır. Çok araçlı yol planlama iki ayrı çerçevede incelenebilir:

- *Sıralı (hiyerarşik) işbirlikçi kontrol*: mantığında merkezi bir karar katmanı bulunur. Bu katman bir grup robot, lider bir robot ya da ayrı bir merkezi karar verici mekanizma olabilir. Bu kontrol yönteminin avantajı çalıştığı çevre hakkındaki bilginin fazlalığıdır.
- *Tepkisel kontrol*: mantığında çevreden gelen uyarılar ya da robotun sahip olduğu algılayıcılar üzerinden alınan bilgiler doğrultusunda robotların kontrolü sağlanır.



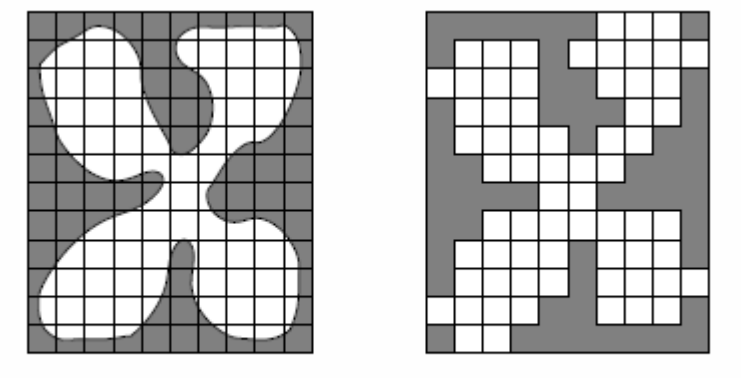
Şekil 2.2 Yol planlamasında sıkça karşılaşılan geçiş sorunu

Çok robotlu bir durumda, hedefler karşılıklı robotlar tarafından şekil 2.2'de olduğu gibi engelleniyorsa en basit yaklaşım, yoluna çıkan diğer robotu ihmal etmektir. Fakat bu yaklaşım robotların çarpışmasına sebep olabilir. Bu tarz bir durumda uygun yaklaşım takımı birlikte modellemektir. Bir robotun diğerine göre geçiş üstünlüğü olmadır. Hangisinin olacağı önceden sisteme girilmişte olabilir, yapılan işin önceliğine ya da robotların pil/yakıt durumları dikkate alınarak seçilebilir. Önceliği olan robot tüm takım tarafından bilinmelidir. Koordinasyon, çok robotlu yol bulma ve planlamada hayati öneme sahiptir.

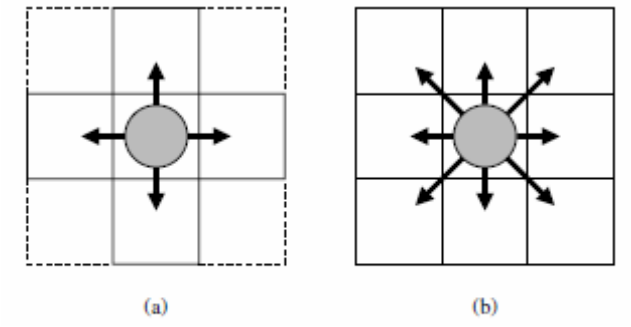
Yol bulma algoritmalarında, çevre bilgisayar bilimindeki çizge kuramı ile tanımlanabilir. Çevreyi tanımlarken bölümlenen koordinatlar kolaylık sağlaması için kare olarak düşünülmüştür. Bir çizge "Ç" iki adet kümeden oluşmuştur Ç=(U,K):

- *Uçlar*: n boyutlu uzaydaki her bir noktaların kümesi
- *Köşeler*: uçları birbirine yönlü ya da yönsüz olarak bağlayan bağların kümesi

Örneğin harita çizge olarak düşünülürse; şehirler uçlar, yollar ise köşeler olur. Benzer bir örnek olarak trafik akışını da verebiliriz. Köşeler yolları, uçlar kesişen yolları, maliyetler hız sınırlarını, tali yolları ya da dar sokakları temsil eder. En kısa ya da en az sıkışık yoldan gitmeye çalışır. Bu yapısal tanımların yanında köşelerin uzunluk ya da maliyet değerlerinin de bilinmesi gerekmektedir. Genelde bu değerler reel sayılar cinsinden ifade edilir. Çizge kuramından faydalanarak, işlem kolaylığı için haritalar belirli en küçük parçalara bölünür. Bu parçalar kare, altıgen ya da üçgen şeklinde olabilir. En çok tercih edilen karelere ayrılmış haritalardır ve mobil araç kontrolünden bilgisayar oyunlarına kadar birçok alanda kullanılırlar. Örneğin şekil 2.3'de bir harita ve onun kareli olarak dönüştürülmüş hali görülebilir. Bu tarz haritalarda hareketli nesnelerin 4 ya da 8 bağlanırlıkları bulunur ve bu yeteneklerine göre yollar çıkarılır (bkz. Şekil 2.4).



Şekil 2.3 Kareli harita çevrim örneği (Smed,Hakonen,2006)

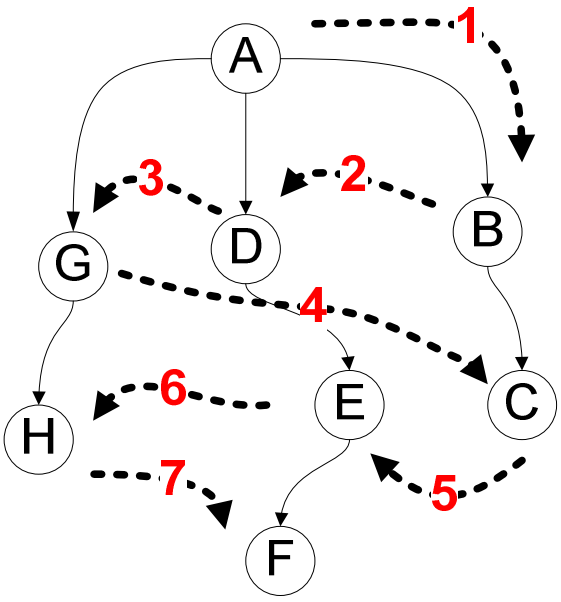


Şekil 2.4 (A) 4 bağlanırlıklı ve (B) 8 bağlanırlıklı kareli harita örneği (Smed,Hakonen,2006)

Her hangi bir yol bulma algoritmasında, yolun tamamen çıkarılması, bir sonraki hareketin seçimi ve öngörülü olması beklenir. Genelde kullanılan yol bulma algoritmaları:

- *Enine arama*: Başlangıç noktasından hedef noktaya kadar olan arama, katmanlar boyunca sürer ve bitimiyle bir alt katmanda arama devam eder. Dijkstra ile benzer olsa da kullanılan yol maliyetlerini dikkate almaz.
- *Derine arama*: Yol maliyetini ve en kısa yol bulmayı hedeflemese de birçok yol bulma algoritmasının esin kaynağı olmuştur.
- *Dijkstra*: En kısa yolu bulma algoritmalarının öncülerindendir, bir dönüm noktası olarak kabul edilir.
- *A**: Hedef duruma olan yolun sezgisel değerlendirmesi yaparak hedefe ulaşmaya çalışır.

Enine arama algoritması, haritada bulunan engelleri ihmal ederek, başlangıç noktasından hedef noktasını, komşu hücrelere dallanarak bulmayı amaçlar. Başlangıç hücresinden itibaren, komşu hücreler, karşılaşma sıralarıyla "açık" adıyla tanımlanan bir listeye eklenir. Açık liste işlenmemiş noktaların saklandığı bir listedir. Bu liste her güncellendiğinde, son eklenen noktanın hedef nokta olup olmadığı kontrol edilir. Hedef nokta bulunmadığı takdirde komşu hücrelere doğru tekrar dallanma gerçekleşir. Başlangıç noktasının maliyet değeri 0, diğer noktaların maliyet değeri ise istenen sabit bir değer olarak atanır. Komşu noktalara her ulaşıldığında kendi maliyet değeri ve ulaşılan noktanın maliyet değeri toplanarak yeni maliyet değeri oluşturulur. Bu değerler daha sonra bulunması durumunda yolun tespitinde kullanılmak üzere maliyet matrisine eklenir. Açık listeye önceden eklenmiş ve kontrol edilmiş noktalar her adımda çıkarılarak aynı noktanın tekrardan işleme alınmaması sağlanır. Eğer açık listenin eleman sayısı sıfıra eşit olursa, verilen başlangıç noktasından hedef noktaya giden uygun bir yol bulunamamıştır. Hedef nokta, açık listeye eklendiği takdirde, en kısa yol, maliyet matrisi üzerinden başlangıç noktasına kadar maliyetleri takip ederek bulunur. Dezavantajı, büyük haritalarda uygun tüm noktaları tek tek kontrol etmesi sonucunda meydana gelen hesap yoğunluğudur. Ayrıca komşu noktalara farklı maliyet değerleri girilemediğinden gerçek dünyanın kısıtlı aktarımı mümkündür (Yüksel, Sezgin, 2006).

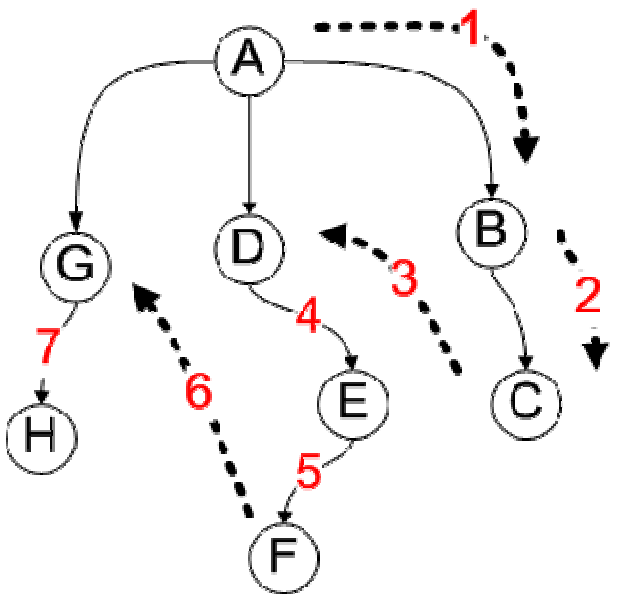


Şekil 2.5 Enine arama algoritması

```
1. AÇIK ve KAPALI ismi ile iki boş liste
   oluştur.
2. Başlangıç noktasını AÇIK listesine ekle.
3. AÇIK listesi dolu olduğu sürece döngüyü
   sürdür:
      a. AÇIK listesindeki ilk elemanı seç ve
         KAPALI listeye taşı.
      b. Eğer seçilen nokta hedef noktası ise;
         döngüden çık ve yolu tanımlayan KAPALI
         listeyi görüntüle.
      c. Eğer seçilen nokta hedef noktası değil
         ise; komşularını sırasıyla AÇIK
         listesinin sonuna ekle.
```

Kod taslağı 2.1 Enine arama algoritması

Derine arama algoritması, enine arama algoritmasının aksine, katman sırasıyla arama yapmak yerine, başlangıç noktasından sonra ilk karşılaşılan noktadan aramaya devam eder. Hedef nokta bulunmadığı takdirde, kaynak noktanın bir diğer komşusunun en derinine inerek aramaya devam eder. Yol bulma uygulamaları esnasında sonsuz bir döngüye takılmaması için algoritmanın gezdiği noktaları etiketlemesi gerekmektedir.



Şekil 2.6 Derine arama algoritması

```
1. AÇIK ve KAPALI ismi ile iki boş liste
   oluştur.
2. Başlangıç ve hedef noktasını tanımla.
3. Başlangıç noktasını AÇIK listesine ekle.
4. AÇIK listesi dolu olduğu sürece döngüyü
   sürdür:
      a. AÇIK listesindeki ilk elemanı seç ve
         KAPALI listeye taşı.
      b. Eğer seçilen nokta hedef noktası ise;
         döngüden çık ve yolu tanımlayan KAPALI
         listeyi görüntüle.
      c. Eğer seçilen nokta hedef noktası değil
         ise; komşularını sırasıyla AÇIK
         listesinin başına ekle.
```

Kod taslağı 2.2 Derine arama algoritması

E. W. Dijkstra'nın 1959 yılındaki "A Note on Two Problems in Connexion with Graphs" (Dijkstra, 1959) bildirisi, yol bulma konusunda çok kullanılan bir referans haline gelmiştir. Dijkstra'nın yol bulma algoritmasını ilgilendiren problem, verilen iki nokta arasındaki en az uzunluğa sahip yolu bulmaktı. Ayrıca bu problem tüm ağırlık değerli grafiksel yol bulma algoritmalarının temelini oluşturmaktadır.

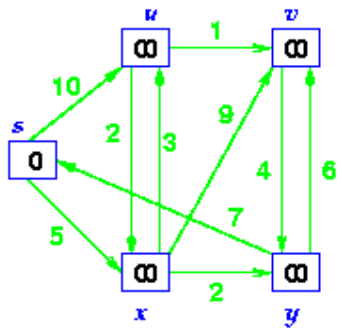
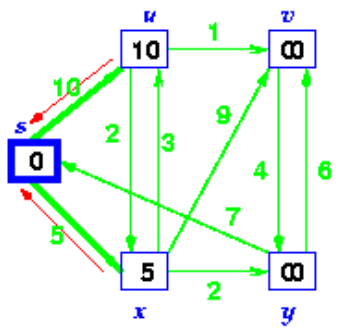
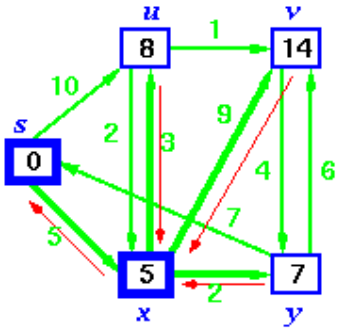
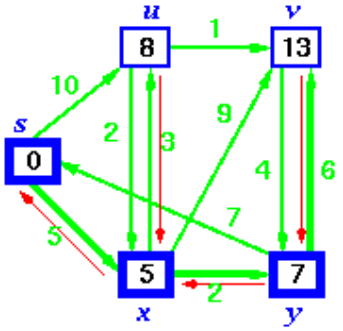
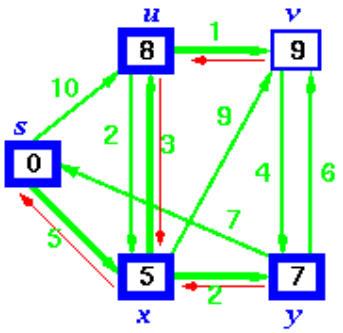
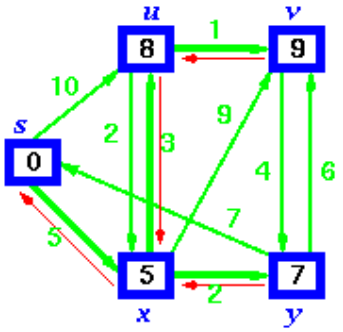
Avantajı, eğer başlangıç noktası ile hedef noktası arasında bir yol varsa bunun en kısa olan değerini göndermesiydi. En kısa yolu, en kısa alt yolları bularak sağlamaktadır. Fakat dezavantajı, tüm noktaların gidilebilir yol olması durumunda, algoritma çok yavaşlamaktaydı (Sidran, 2005).

```
1. Başlangıç ve hedef noktasını tanımla
2. Harita matrisini yükle
3. Başlangıç noktasını AÇIK listeye ekle
4. Komşularını AÇIK listeye ekle, maliyetlerini hesapla,
   kaynak noktayı KAYNAK listesine ekle.
5. Eğer AÇIK liste boş ise, yol bulunamamıştır.
6. Eğer hedef nokta AÇIK listeye eklendi ise, KAYNAK
   listesini kullanarak yolu tanımla.
        Değil ise devam et.
7. Eğer önceden AÇIK listede olan komşu noktalar var
   ise, yeni maliyetlerini hesapla ve eskileri ile
   karşılaştır. Eğer daha düşük ise, maliyeti ve KAYNAK
   listesini güncelle.
8. İşlemi biten noktayı AÇIK listeden kaldır.
9. 4. Adıma geri dön.
```

Kod taslağı 2.3 Dijkstra algoritması (Yüksel, Sezgin, 2006)

Dijkstra algoritmasının, en kısa yol ile birlikte en düşük maliyetli yolu bulma yeteneği bulunmaktadır. Algoritma, çalışmak için açık ve kaynak listesine sahiptir. Öncelikle başlangıç noktasının komşuları, açık listeye eklenir. Başlangıç hücresinden, komşu hücrelere olan maliyetler hesaplanır. Maliyetler, artan sıra ile açık listede tekrardan sıralanır. Komşu noktaları eklenen kaynak noktaları, bulunması halinde yolu belirlemek için kaynak listesine eklenir. Ek olarak eğer komşu noktanın maliyeti kaynağından daha düşükse, komşu nokta kaynak olur ve önceden bu kaynak nokta tarafından eklenmiş komşuların maliyeti tekrardan hesaplanır. Bu hesaplamalar ve yerleştirmeler, hedef noktanın açık listeye eklenmesine kadar ya da açık listenin hiç elemanı kalmayana kadar sürer. Hedef nokta açık listeye eklendiği takdirde, kaynak listesi dikkate alınarak aranan yol bulunmuş olur (Yüksel, Sezgin, 2006). Algoritma, tablo 2.6'da adım adım bir örnek ile açıklanmaya çalışılmıştır.

Tablo 2.6 Dijkstra algoritmasının uygulama örneği (Morris,1998)

<table>
<tr><td> 
</td><td> 
</td></tr>
<tr><td>Başlangıç (s) ve hedef noktası (v) seçilir. Maliyet matrisi yüklenir.</td><td>Başlangıç noktasına en yakın nokta seçilir, maliyetler hesaplanır ve kaynak listesine eklenir.</td></tr>
<tr><td> 
</td><td> 
</td></tr>
<tr><td colspan="2">Başlangıç noktasının komşularına yayılıp, maliyetler güncellenir ve kaynak listesine eklenir.</td></tr>
<tr><td> 
</td><td> 
</td></tr>
<tr><td>Tekrardan en yakın nokta seçilir, maliyetler ve kaynak listesi güncellenir.</td><td>Hedef nokta kaynak listesine eklendikten sonra, liste takip edilerek yol bulunur.</td></tr>
</table>

1968 yılında, Hart, Nilsson ve Raphael'in “A Formal Basis for the Heuristic Determination of Minimum Cost Paths” (Hart,Nilsson,Raphael,1968) bildirisiyle A* algoritması ortaya çıkmıştır.

Dijkstra'ya benzeyen A* algoritması, iki listeye sahipti. Açık listesi genellikle öncelik sırası olarak uygulanır. Kapalı liste, üzerinde işlem yapılmış noktaların saklandığı liste olarak kullanılır.

A* algoritmasının Dijkstra'dan ayrılan yönü, f(n), maliyet fonksiyon yaklaşımı olmuştur. Maliyet fonksiyonunu iki ayrı fonksiyonunun toplamı ile hesaplanmaktaydı. Birinci fonksiyon, g(n), başlangıç noktasından n. nokta arasındaki gerçek hareket maliyet fonksiyonudur. İkicisi, h(n), n. noktadan hedef noktaya olan tahmini maliyet fonksiyonudur (Sidran, 2005). A* algoritmasının kullanımının tercih sebebi, gidilebilecek bir yol olduğu takdirde, her hangi bir başlama noktasından, her hangi bir bitiş noktasına olan en iyi yolu bulmayı garanti etmesidir (Bourg,Seeman,2004).

A* algoritmasında en uygun yolu bulmaya çalışılırken, öncelik sıraları kontrol edilmemiş yolları saklamaktadır. Sıralama en kısa yola en yüksek öncelik verilmesiyle sağlanmaktadır. Öncelik sırası veri yapıları, rastgele aralıklarla yeni eleman girişine izin verdiği için basit sıralama yapılarından daha esneklik sağlamaktadır. Öncelik sırasına yeni eleman eklemek, her ekleme sonrasında yeniden sıralamaktan çok daha az maliyetlidir (Skiena,2008).

```
1. Başlangıç “b” ve hedef “h” noktasını tanımla.
2. “AÇIK” ve “KAPALI” adlı boş listeler tanımla.
3. b’yi AÇIK listesine ekle.
4. G(b) ← 0.
5. H(b)’yi hesapla.
6. F(b) ← G(b) + H(b)
7. AÇIK listesi dolu olduğu sürece döngüyü sürdür:
   a. AÇIK listesindeki en küçük F maliyet fonksiyon
      değerine sahip noktayı “s” seç.
   b. Eğer s, h'ye eşit ise, yolu tanımla.
   c. Değil ise, s'yi AÇIK listesinden KAPALI listesine
      taşı.
   d. s'nin her komşu noktasına “k” uygula:
        – Eğer k KAPALI listede ise, geç.
        – Geçici G(k)'yi, G(s) ve s ile k arasındaki
          uzaklığın toplamına eşitle.
        – Geçici değer iyi değil.
        – Eğer k AÇIK listede değil ise; listeye ekle,
          H(k) hesapla, geçici değer iyidir.
        – Değilse, eğer Geçici G(k), G(k)'den küçük
          ise, geçici değer iyidir.
        – Eğer geçici değer iyi ise; k'nin kaynağını s
          olarak etiketle, G(k) değerine Geçici G(k)
          değerini ata, F(k) ← G(k) + H(k).
```

Kod taslağı 2.4 A* algoritması (Yüksel, Sezgin, 2006)

2.3 Bilgisayarlı Görü

Projede, bilgisayarlı görü sistemlerinden olan genel görü sistemleri seçilmiştir. Bu sayede kameranın görüntü alanı içinde kalan mobil araçlar sürekli gözlenmektedir. Bu gözlem sırasında mobil araçların ve engellerin konumları renk filtresi yardımıyla ortamdan ayrıştırılmaktadır. Ayrıştırılan görüntü, kabarcık bileşenler yöntemi kullanılarak, piksellerin ağırlık merkezi hesaplanarak, mobil araçların takibi sağlanmaktadır. Engeller, haritaya kabarcık alanı olarak yansıtılmaktadır.

2.3.1 Genel Görü Sistemi

Genel görü sistemleri, kullanılan kameranın görüntü alanını içinde, bilgisayarlı görü işlemlerini yapabilecek ortam sunmaktadır. Bu sistemler, hem kendi hem de uygulandıkları diğer sistemlerin (örneğin çok robotlu sistemler) maliyetini düşürdükleri için tercih edilirler. Ayrıca en uygun stratejik planlamayı hem donanımsal olarak hem de sahip oldukları bilgi birikimleri yardımıyla yapabilirler. Dezavantajları, sabit olmalarıdır. Görüntü alanının yetmediği ya da hareketlilik gerektiren durumlarda yerel görü sistemleri kullanılabilir. Bu sayede, her birimin kendi kamera sistemi olur. Fakat kameradan alınan bilgilerin işlenme maliyeti birim sayısı ile orantılı biçimde artacağından maliyete olumsuz etkisi olur. Ayrıca her robot kendisi için belirli fakat benzer görüntü işleme algoritmalarını yürütecekleri için fazladan iş yapmaya sebep olurlar. Genel görü sistemleri, özellikle yapılan robot futbol yarışmalarında avantajları ile ön plana çıkmıştır (Dichiera,1999).

Genel görü sistemlerinin yapısında, kamera, işlemci ve kablosuz bağlantı donanımı yer almaktadır (Perš,Kovacic,2000, Gupta,Messom,Demidenko,2005). Kamera, aldığı her yeni görüntüyü sistemin beyni olan işlemciye yollar ve işlemcide hesaplanan kontrol parametreleri kablosuz bağlantı protokolleri yardımıyla birimlere iletilir.

Şekil 2.7 “Logitech” firmasının “QuickCam Pro for Notebooks” model kamerası

Genel görü sisteminin, görüntü alanı (bkz. Şekil 2.8) ve diğer kalibrasyon değerleri hesabı için temel trigonometrik bilgilere ve kullanılan "Logitech" firmasının "QuickCam Pro for Notebooks" model kamerasının (bkz. Şekil 2.7) teknik verilerine dayanılarak yapılmıştır (Wachi,2009, Twenga,b.t.). Genel görü sistem düzeneği kurulup, kamera yüksekliği, odak uzaklığı, görüntü alanında kalan gerçek uzaklık değerleri (en ve boy) ve alınan görüntü çözünürlüğü (Proje genelinde $640px \times 480px$ kullanılmıştır.) bilgileri (bkz. Tablo 2.7) ile birim piksel değerinin milimetre karşılığı, "*gerçek piksel uzaklığı*" hesaplanmıştır.

$$Gerçek\ piksel\ uzaklığı\ \left(\frac{px}{mm}\right) = \frac{u_{gerçek} \times u_{odak}}{y_{kamera} \times u_{sanal}}$$

Tablo 2.7 Bilinen genel görü sistem değerleri

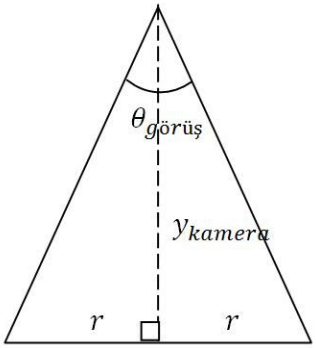
Parametre	**Değer**			
Kamera yüksekliği (y_{kamera})	2,5 m			
Odak uzaklığı (u_{odak})	3,7 mm			
Gerçek uzaklık $(u_{gerçek})$	*Genişlik:*	264 cm	*Yükseklik:*	198 cm
Sanal uzaklık (u_{sanal})		640 px		480 px

Şekil 2.8 Genel görü düzeneğinden alınmış kamera görüntüsü

Bilgiler ışığında, düzenek için elde edilen gerçek piksel uzaklığı 6,105x10^{-3} px/mm olarak bulunmuştur. Ayrıca görüntü yarıçapı ve görüş açısı hesabı için tablo 2.7'deki gerçek boy ve en uzaklıkları kullanılmıştır (bkz. Tablo 2.8).

$$r \cong \sqrt{\frac{u_{gerçek}^{boy} \times u_{gerçek}^{en}}{\pi}}$$

$$Görüş\ açısı\ (\theta_{görüş}) = 2 \times tan^{-1}\left(\frac{r}{y_{kamera}}\right)$$

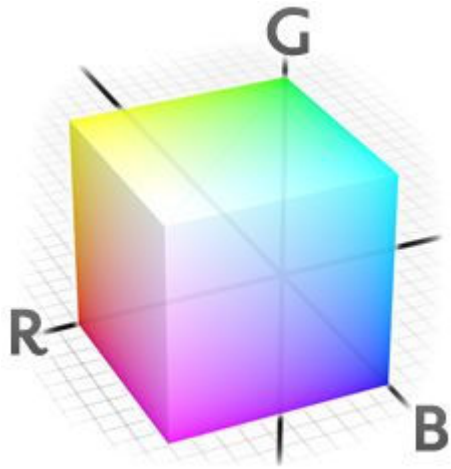


Tablo 2.8 Hesaplanan genel görü sistem değerleri

Parametre	Değer
$Görüntü\ yarıçapı$ (r)	$\sim 1{,}290\ m$
$Görüş\ açısı\ (\theta_{görüş})$	$\sim\ 54{,}58^{o}$

2.3.2 Renk filtreleme

Renk filtrelemenin amacı, belirli bir görüntü için verilen bir eşik değerinin üzerindeki ya da istenen renk aralığındaki renk değerini görüntünün geri kalanında ayırmaktır. Renk filtreleme uygulanırken kullanılan renk modelini dikkate almak gereklidir. Projede kullanılan kamera, RGB (Kırmızı [Red], Yeşil [Green], Mavi [Blue]) renk modelini kullanmakta olup, bu 3 rengin birleşimleri ile görüntüyü sisteme sağlamaktadır. RGB renkleri [0,255] aralığı ile tanımlanmaktadır. Örneğin tam kırmızı rengin RGB değer karşılığı (255,0,0), sarı rengin RGB değer karşılığı (255,255,0) ve beyaz renginin RGB değer karşılığı ise (255,255,255) olarak tanımlanır. Bu ve diğer renk değerleri, şekil 2.9'deki RGB renk model koordinat eksenlerinden de çıkarılabilir. Ayrılmış görüntü genelde siyah beyaz örneğinde olduğu gibi ikili resim olarak saklanır. Bu sayede görüntüyü, kabarcık analiz yöntemi gibi diğer bilgisayarlı görü algoritmalarının işlemesi için uygun hale getirmiş olur.



Şekil 2.9 RGB renk model koordinat ekseni
(Color Models,b.t.)

RGB renk modeline göre renk filtreme yapmak için öncelikte görüntüyü kamera yardımıyla hafızaya alınır. Filtrelenmesini istediğimiz rengin RGB renk aralığı algoritmaya verilir. Algoritma, hafızadaki görüntünün her piksel değerinin RGB renk aralığı içinde bulunup bulunmadığını kontrol eder. Aralık içinde kalan pikseller beyaz renk ya da kendi mutlak rengi ile değiştirilir. Bu koşula uymayan diğer tüm pikseller siyah olarak değiştirilerek, renk filtresi uygulanmış, görüntüde ikili görüntü haline getirilmiştir. Görüntü çözünürlüğünün büyük olması durumunda ya da kullanılan bilgisayarın işlem performansını arttırmak gerektiğinde arama belirli bir pencere ile sınırlandırılabilir. Eğer bu pencere içinde ilgilenilen pikseller bulunmadıysa pencere genişletilebilir veya performans artırımı göz ardı edilerek tüm piksel değerleri araştırılabilir.

```
1. Görüntüyü al.
2. İstenen kırmızı, yeşil ve mavi renk aralıklarını
   belirle.
3. Görüntünün her pikseline git:
      a. Bulunduğu "β" pikselinin; kırmızı, yeşil ve
         mavi renk değerlerini al.
      b. Eğer
         (β.Kırmızı ≥ Kırmızı.Alt VE β.Kırmızı ≤ Kırmızı.Üst) VE
         (β.Yeşil ≥ Yeşil.Alt VE β.Yeşil ≤ Yeşil.Üst) VE
         (β.Mavi ≥ Mavi.Alt VE β.Mavi ≤ Mavi.Üst)
         ise β'nin değerini aranan renk değerine eşitle.
      c. Değil ise, β'yi siyaha eşitle.
```

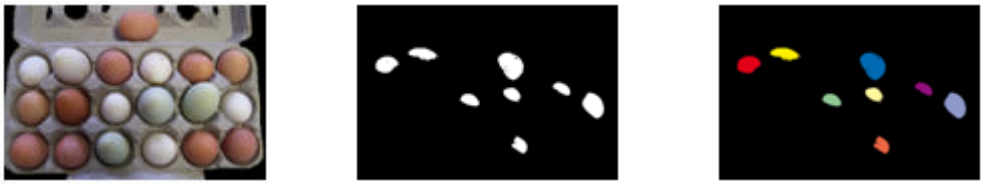
Kod taslağı 2.5 Renk filtreleme algoritması

Renk filtrelenmesi için aralık değerlerine bakılmanın dışında, aranan (α) renk ile arama sırasında bulunulan (β) renk değerinin "Euclidian" mesafesi belirli bir aralık ile de kontrol edilebilir.

$$\sqrt{(\beta_{kırmızı} - \alpha_{kırmızı})^2 \times (\beta_{yeşil} - \alpha_{yeşil})^2 \times (\beta_{mavi} - \alpha_{mavi})^2} \leq Eşik$$

2.3.3 Kabarcık Analiz Yöntemi

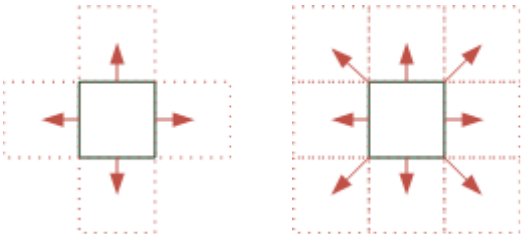
Birbirine bağlı piksellerin oluşturduğu gruba, kabarcık adı verilir. Genellikle kabarcıklar belirli bir filtre işlemi sonucunda elde edilir. Örneğin basit bir eşik değeri filtresi ile verilen eşik değerinin altın kalan pikselleri arka plan rengi, üzerinde olanlar ise kabarcık olarak tanımlanır. Bu ayırma işlemi sonrasında, bağlı pikselleri etiketleyerek, kabarcıkları kendi aralarında ayrılır (bkz. Şekil 2.10).



İşlenmemiş Resim Eşik değer filtreli Resim Etiketlenmiş Resim

Şekil 2.10 Kabarcık analiz yönteminin adımları (NGI Book,2007)

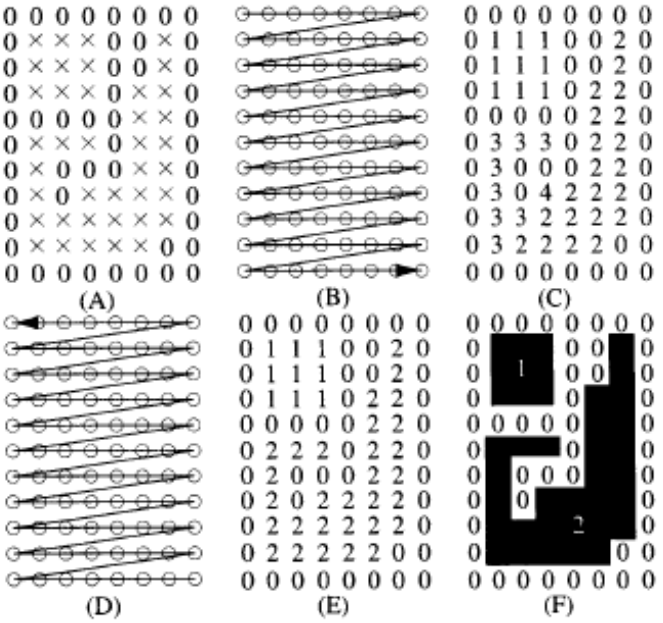
Genellikle kabarcıklar filtrelenirken pikselleri, arka plan pikselleri (0 değerli) ve ön plan pikselleri (0 harici değer) olmak üzere iki kategoriye ayırır. 4'lü ya da 8'li bağlanırlık (bkz. Şekil 2.11) değerler seçimine göre, kabarcık sayısı değişiklik gösterir. 8'li bağlanırlığın seçimi ön plan pikselleri, 4'lü bağlanırlığın seçimi ise arka plan piksellerinin filtrelenmesi için faydalıdır.



4'lü Bağlanırlık 8'li Bağlanırlık

Şekil 2.11 Piksellerin bağlanırlık durumları

Görüntü üzerinde kabarcık analizine başlamadan önce, görüntü renk filtresinden geçirilir (bkz. Bölüm 2.3.2). Renk filtresi, ilgilenilecek renge sahip cisimlerin ön plana çıkması sağlar. Renk filtresi sonucunda elde edilen siyah-beyaz resim bağlı bileşenler yöntemi ile kabarcık analizine başlamak üzere hafızaya alınır. Araştırma sonuçlarını saklamak için oluşturulan "etiket" listesi oluşturulur ve sıfırlanır. Görüntüdeki her beyaz piksel değerine, komşularının sahip olduğu en küçük etiket değerini ata. Eğer komşularının etiket değeri bulunmuyorsa, etiket listesindeki son değeri bir arttırıp, pikselin etiketi olarak belirle. Görüntünün son piksel değeri $\left((x_{son}, y_{son}) = \left(görüntü_{boy} - 1,\ görüntü_{en} - 1\right)\right)$ etiketlendikten sonra fazladan etiketlenme olabileceği için bu noktadan geriye, (0,0) pikseline, etiket listesi komşuların en küçüğü seçilerek güncellenir (bkz. Şekil 2.12).

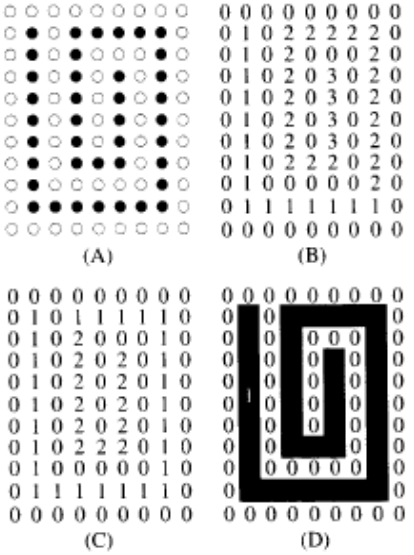


Şekil 2.12 Bağlı bileşenler yöntemi ile kabarcıkların taranma adımları: (A) Siyah (0)-beyaz (x) resim alınır, (B) Baştan sona kadar taranır, (C) Komşularının etiketi olmadı takdirde etiket listesindeki son değer + 1, kendi etiket değeri olur, (D) En son pikselde etiketlendikten sonra, başa doğru tekrar tarama başlar, (E) İkinci tarama yardımıyla fazladan etiketlenen piksel değerleri güncellenir, (F) Birbirine bağlı cisimlerin analizi tamamlanır. (Marchand-Maillet,Sharaiha,2000)

1. İkili resmi hafızaya al.
2. Etiket listesini sıfırla.
3. Resimdeki her piksel değerine baştan başlayarak uğra:
4. Eğer uğranan piksel değeri siyah değil ise;
    a. Eğer piksel değerinin komşuları etiketlenmemiş ise; Etiket listesindeki değeri bir arttır ve pikseli etiketle.
    b. Eğer piksel değerinin komşuları etiketlenmiş ise; en küçük etiket değerini, pikselin etiketi olarak ata.
5. Resimdeki her piksel değerine sondan başlayarak uğra:
    a. Eğer uğranan pikselin etiketi, her hangi bir komşusunun etiketinden büyük ise; uğranan pikselin etiketini en küçük komşu piksel etiketi ile değiştir.

Kod taslağı 2.6 Bağlı bileşenler yöntemi ile kabarcık çıkarım algoritması

Kod taslağı 2.6'daki algoritmada olduğu gibi, görüntünün sahip olduğu tüm pikselleri bir baştan bir de sonra tarayarak en kötü senaryoda (bkz. Şekil 2.12) bile bağlı bileşenlerin doğru ve yerinde olması sağlanmaktadır. En kötü senaryoda sarmal biçimde yer alan cisim ilk arama esnasında fazladan etiketlenmektedir.



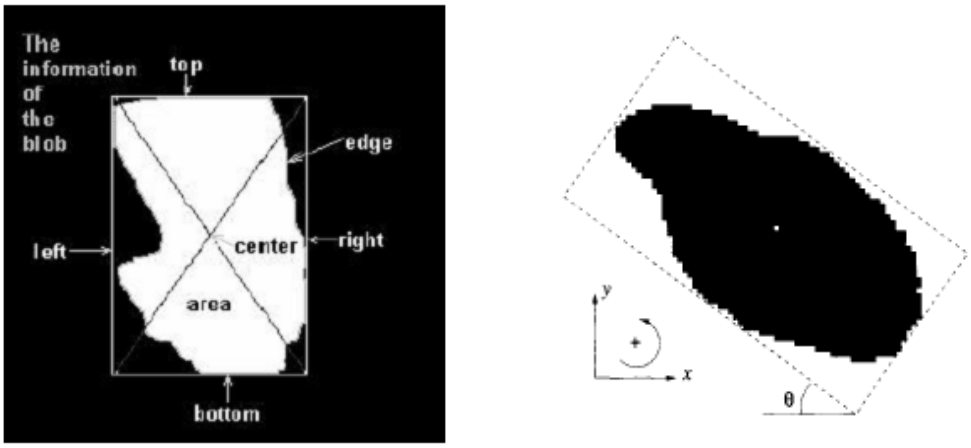
Şekil 2.12 Bağlı bileşenler yönteminde karşılaşılabilecek en kötü durum senaryo örneği: (A) Hafızadaki görüntü, (B) İlk etiketleme adımından sonraki durum (C) İkinci etiketlemenin başlamasıyla gözlenen iyileşmeler ve (C) Sonuç.(Marchand-Maillet,Sharaiha,2000)

Kabarcıkların bağlı bileşenler yöntemi ile bulunması sırasında alan, ağırlık merkezi ve moment, dolayısıyla rotasyon, bilgileri de alınabilir. Alan bilgisi basitçe üzerinde geçilen ve siyah olmayan pikselleri sayarak hesaplanabilir. Ağırlık merkezinin x ve y koordinatları, ilgili koordinat bileşeninin sayısının alana bölümü olarak tablo 2.9 (A) ve (B)'deki denklemlerle hesaplanabilir. Bir görüntünün momenti, görüntünün piksel yoğunluğunun ağırlıklı ortalaması olarak tanımlanır ve şekle özgü özellikleri tanımlar. (k,l)'inci merkezsel moment genellemesi kabarcıkların rotasyonu yanında daha önceden de bahsedilen alan ve ağırlık merkezi bulmada kullanılabilir. Bu hesaplamalarda tablo 2.9 (C)'deki denklem kullanılarak 0'ıncı merkezsel momentin kullanımı alanı, 1'inci merkezsel momentin kullanımı ağırlık merkezini ve 2'inci merkezsel momentin kullanımı rotasyonu vermektedir.

Tablo 2.9 Kabarcık özelliklerinin hesabı: **(A)** Ağırlık merkezinin x ve **(B)** y bileşeni, **(C)** Merkezsel moment, **(D)** Rotasyon açısı.

(A)	$x_m = \frac{1}{genişlik} \sum_{i=1}^{genişlik} x_i$	**(B)**	$y_m = \frac{1}{yükseklik} \sum_{j=1}^{yükseklik} y_j$
(C)	$\mu_{k,l} = \sum_{i=1}^{n} (x_i - x_m)^k (y_i - y_m)^l$	**(D)**	$\theta = \frac{1}{2} \arctan(\frac{2\mu_{1,1}}{\mu_{2,0} - \mu_{0,2}})$

Ayrıca kabarcığı en dıştan çevreleyen dikdörtgenin, dış sınırın, uzunluk bilgileri kabarcığın analizinde sıklıkla kullanılmaktadır.



Şekil 2.14 Kabarcık özelliklerinin grafiksel gösterimi (Marchand-Maillet,Sharaiha,2000)

BÖLÜM ÜÇ
"ÇOK SİLAHŞORLAR" YAZILIM PROJESİNE GENEL BAKIŞ

3.1 Tanıtım

"Çok Silahşorlar" yazılım projesi, merkezi mimari modeli ile tasarlanmış çok ajanlı sistemleri esas almaktadır. Mobil araçlardan oluşan ajanlar, kesin olarak iletişim halindedir. Mobil araçlar, stratejik oyun modeline dayanan karar verme mekanizmaları ile otonom ve işbirliği içindedirler. Davranış tabanlı görevler mobil araçlara tek tek dağıtılmaktadır.

"Çok Silahşorlar" yazılım projesi, görüntü alanı içinde bulunan, kendi renk etiketlerine sahip mobil araçları, birbirlerine ve belirli renge sahip engellere çarpmadan, kullanıcı tarafından istenilen noktaya gidebilecek en kısa yoldan sürmektedir. Belirli bir görüntü alanını oluşturmak ve taşınabilir olmasını sağlamak amacıyla şekil 3.1'deki düzenek hazırlanmıştır. Bu düzeneğin yüksekliği 2,7 m, genişliği 2,5 m'dir. Düzeneğin tam ortasında bulunan platforma, kamera tutturulmaktadır. Kullanılan kameranın sistem performansını ve güvenilirliğini doğrudan etkileyeceği göz önünde tutularak, en iyi gerçek çözünürlüğe sahip olan "Logitech" firmasının "QuickCam Pro for Notebooks" (bkz. Şekil 2.7) modeli kullanılmıştır. Genel görü sistemleri ile kullanılacak mobil araçların kablosuz iletişim ihtiyaçları bulunmaktadır. Bu ihtiyacı karşılarken gerek yazılımcı gerek kullanıcı tarafının işlem maliyetini azaltmak amacıyla, günümüzde yaygın kullanılan "Bluetooth" kablosuz iletişim protokolünü destekleyen, "Lego Mindstorms NXT" setleri kullanılmıştır. Bu setlerin yardımıyla farklı tasarımlara sahip mobil araçlar denenmiş, istikrarlı ve az parça gereksinimi olan tasarımlar seçilmiştir.

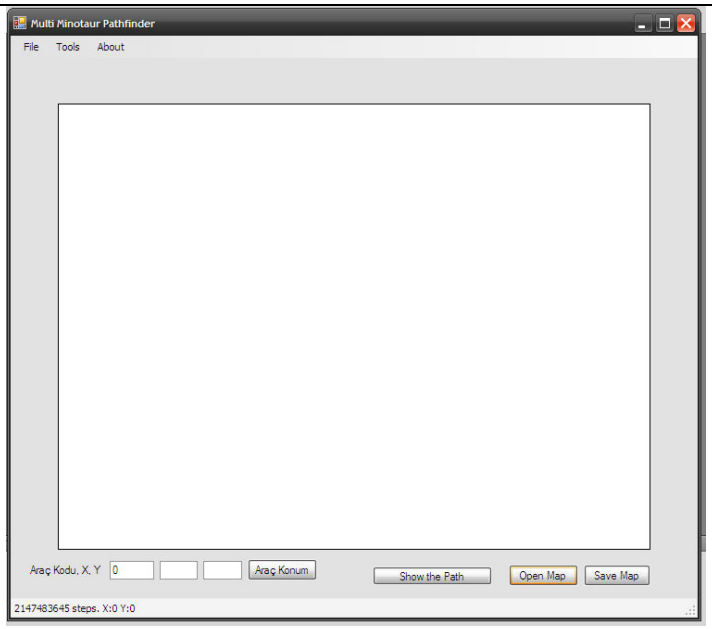
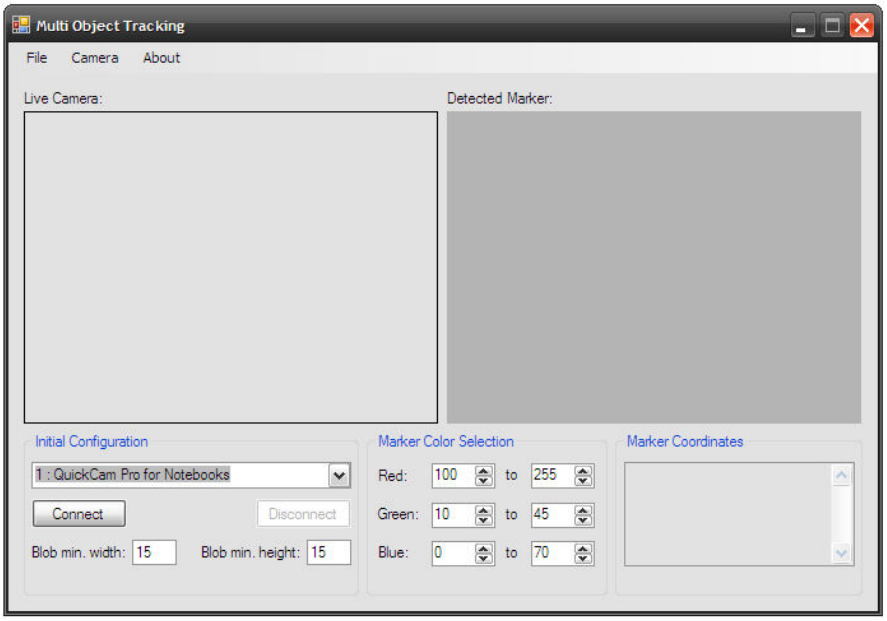
Proje hazırlanırken bir kontrol programının yanı sıra, benzer çalışmaları da desteklemek amacı güdülerek, bir yazılım çerçevesi olarak tasarlanmıştır. Bu tarz büyük bir programları geliştirmek için yazılım mühendisliği deneyimlerinden faydalanarak, küçük program parçaları halinde hazırlanıp, birleştirilmiştir. Bu şekilde hazırlanan yazılımların hata tespiti ve değişen şartlara göre geliştirmesi sağlanmıştır.

Yazılım; bilgisayarlı görü, yol bulma ve araç kontrol parçaları olmak üzere 3'e ayrılmıştır.

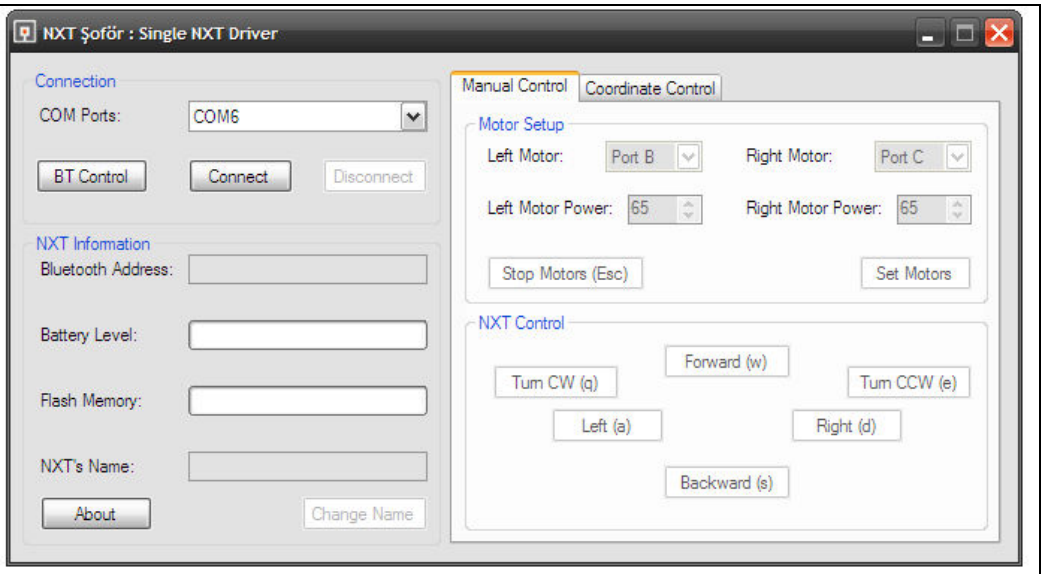
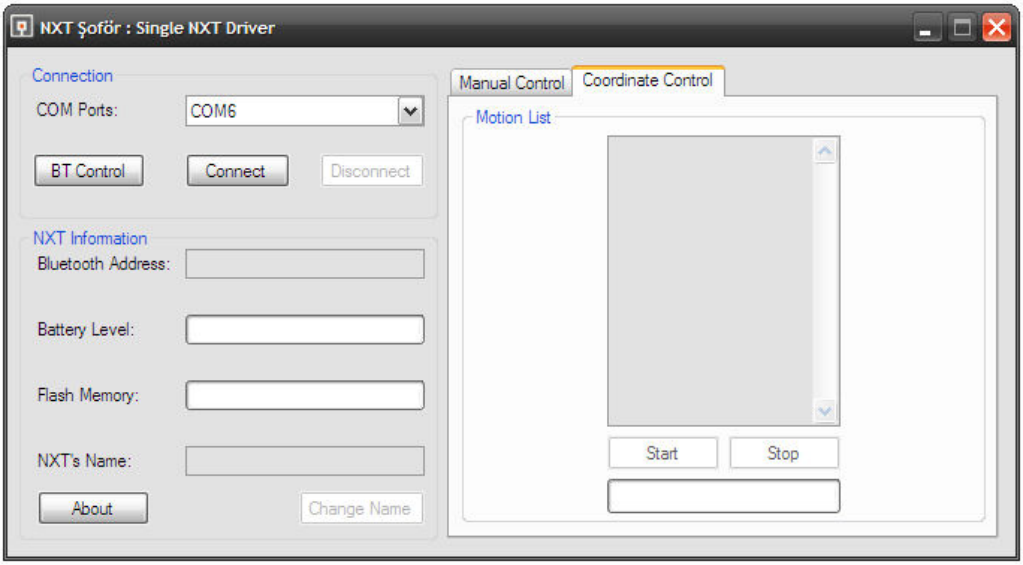
Şekil 3.1 Genel görü sistemi için hazırlanan düzenek

Proje kapsamında hazırlanan alt program modülleri tek başlarına da benzer görevlerde kullanılabilmek üzere kendilerine özel arayüz ile kullanıma sunulmuştur. Alt program modülleri; (a) çoklu renk takibi, (b) çok araçlı yol bulma ve planlama ve (c) NXT sürücüsünden oluşmaktadır. Tüm bu alt modüller bir birlerinden bağımsız olarak geliştirilmiş ve testlerden geçirildikten sonra "Çok Silahşorlar" ana programı tarafından birleştirilerek görüntü alanı içinde çoklu mobil araçların kontrolünü sağlamakta kullanılmıştır.

Tablo 3.1 Alt program modülleri ve açıklamaları

Alt program modülü	
Açıklaması	Çok araçlı yol bulma ve planlama alt programının amacı elle girilen iki konum çifti (başlangıç-bitiş) arasındaki en uygun yolu engellerden de sıyrılarak bulmaktır. Alt programın başlangıç noktaları elle, bitiş noktaları ise fare ile girilmektedir. Engeller yine fare yardımıyla çizilebilir. Meydana getirilen harita metin dosyası olarak ileride çalışmaya devam etmek üzere saklanıp tekrardan açılabilir.
Alt program modülü	
Açıklaması	Çoklu renk takibi alt programının amacı belirlenen renk aralığına sahip üç noktanın koordinatlarını ayrı ayrı takip edip listelemektedir. Renkli noktalar renk filtresi ile çevresinden ayırt edilip, kabarcık analiz yöntemi ile ağırlık merkezlerinin koordinatı hesaplanmaktadır.

Tablo 3.1 Alt program modülleri ve açıklamaları (devamı)

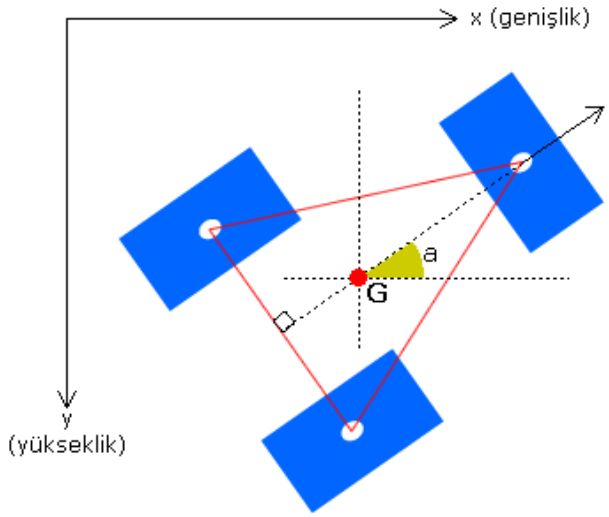
Alt program modülü	
Açıklaması	NXT sürücüsü alt programının iki amacı vardır. Birincisi bluetooth seri iletişim bağlantı noktası ile bağlanan aracın donanım adresini, pil seviyesini, hafıza durumunu ve ismini gösterir. Cihazın ismi istenildiğinde değiştirilebilmeye de olanak sağlar. İkinci amacı ise mobil aracı elle kumanda edebilmesidir. Elle kumanda için ilgili yön düğmeleri ya da klavye kısa yolları kullanılabilir.
Alt program modülü	
Açıklaması	NXT sürücü programının elle sürüş seçeneği yanında hareket listesini takip ederek sürme seçeneği de bulunmaktadır. Bu liste ileride bahsedilecek olan "Çok Silahşorlar" ana programının arayüzünden oluşturulabilir. Liste istenildiğinde elle de yazılabilir. (örneğin, ileri için "1,0", geri için "-1,0", saat yönünde dönüş için "0,1" ve saat yönünün tersi için "0,-1")

3.2 Çalışma Mantığı

"Çok Silahşorlar" yazılım projesi, başlangıç ayarlarında sonra, sadece kameradan aldığı görüntü bilgisini temel alarak çalışmaktadır (bkz. Şekil 3.3). Başlangıç ayarları:

- *Mobil araçların bağlantı girişi*: Araçların kullanılacak bilgisayar ile "Bluetooth Seri Bağlantı Girişi" üzerinden eşleştirilir. Bu işlem, kullanılan araçlar için bir defaya mahsus olarak yapılması yeterlidir. Sisteme dâhil edilen yeni araçların olması durumunda sadece bu araçların eşlemesi yapılır.
- *Mobil araçların etiketleri*: Her araca kendine ait, farklı bir renk seçilir. Bu renkler istenildiği gibi seçilebilir. Seçilen renkler, ileride karışıklık olmaması ve kullanım kolaylığı için araçların iç hafızalarına yazılır.
- *Engel etiketi*: Görüntü alanında bulunan engellerin ortak rengidir.
- *Hedef noktaları*: Her araca, gitmesi istenen nokta verilir.
- *Öncelik sırası*: Araçların hangi özelliklerine göre önceliğe sahip olacakları seçilir

Bu ayarların girilmesi ile araçların bulundukları konum, görüntü üzerinden renk filtreleme algoritması ve ardından kabarcık analizi ile hesaplanır.

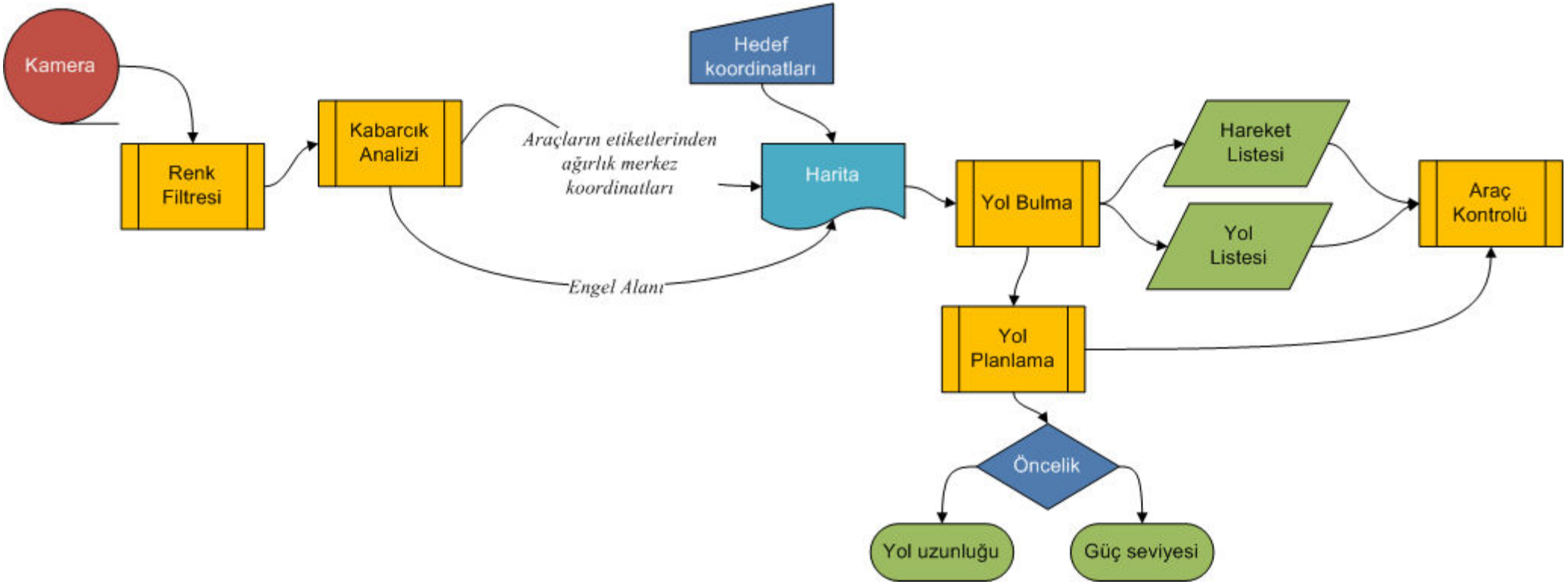


Şekil 3.2 Mobil aracın renk etiket dağılımı, "G" ağırlık merkezi,"a" hesaplanan açı değeri ve doğrultusu

Araçların üzerinde 3 adet aynı renkten etiket bulunmaktadır. Bu sayede piksel koordinatlarına göre konumu ve doğrultusu hesaplanabilir. Aracın konumunu bulmak için öncelikle 3 etiketin orta noktası kabarcık analizi yöntemi yardımı ile hesaplanır. Etiketlerden en yakın konuma sahip 2 tanesi üçgenin taban kenarlarını, geriye kalan nokta ise üçgenin tepe kenarını oluşturur (bkz. Şekil 3.2). Bu sayede elde edilen üçgen, ikizkenar olarak kabul edilip, bu kabule göre ağırlık merkezi ve tepe açısı geometrik bağıntılar yardımıyla hesaplanır. Etiketlerden oluşturulan ikizkenar üçgenin ağırlık merkezi mobil aracın konumunu, tepe açısının yarısının piksel koordinat düzemi ile yaptığı açı değeri ise doğrultusunu verir. Bu sayede mobil araçların hedeflerine ulaşmaları için hesaplanan yollarda uygun bir şekilde sürülmeleri sağlanmaktadır. Uygun sürüşü sağlamak ve fazladan yol aldırmamak amacı ile araçların belli açı aralıklarında saat yönünde, diğer açı aralıklarında ise saat yönünün tersinde dönmeleri uygulanan algoritma ile sağlanmıştır.

Engeller, rengine göre filtrelenip kabarcık analizi ile alanı hesaplandıktan sonra haritaya aktarılır. Kabarcık analizi sonucunda en büyük alana sahip kabarcık sistemin engeli olarak kabul edilir. Engel haritaya aktarılırken daha hızlı ve toleranslı olması amacı ile “Gauss” yöntemi ile bulanıklaştırma ve sonucu genleştirme yöntemleri de kullanılmıştır. “Gauss” yöntemi ile bulanıklaştırmanın temeli alçak geçiren filtreye dayanmaktadır. Genleştirme sayesinde üzerinde çalışılan görüntünün çevresi genişletilerek ve yumuşatılarak araçların çarpma ihtimali azaltılır.

Sistem artık en kısa yolu bulmak için hazırdır. Her araç için yol bulma işlemi sonunda, hareket ve yol listesi adında iki liste oluşturulur. Hareket listesi, aracın hangi hareketleri yaparak görevini tamamlayabileceğini bulundurur. Yol listesi ise hedef noktaya giderken geçeceği koordinatlara sahiptir. Her aracın bulunduğu konumdan, hedef konumuna olan yolu, saniyede 10 defa, sistem performansını etkilemeden hesaplanır, hareket ve yol listesi güncellenir. Bu sayede araçların, diğer araçlarla karşılaşma ihtimali azaltılır, engellerin yer değiştirme durumuna karşılık çarpışmaları engeller. Güncellemelerin yetersiz olduğu durumlarda, araçlar belirlenen öncelik sırasına göre hareket ederler. Bu sıra, yolu en uzun araca veya pil seviyesi en az olan araca yol vermek olabilir.

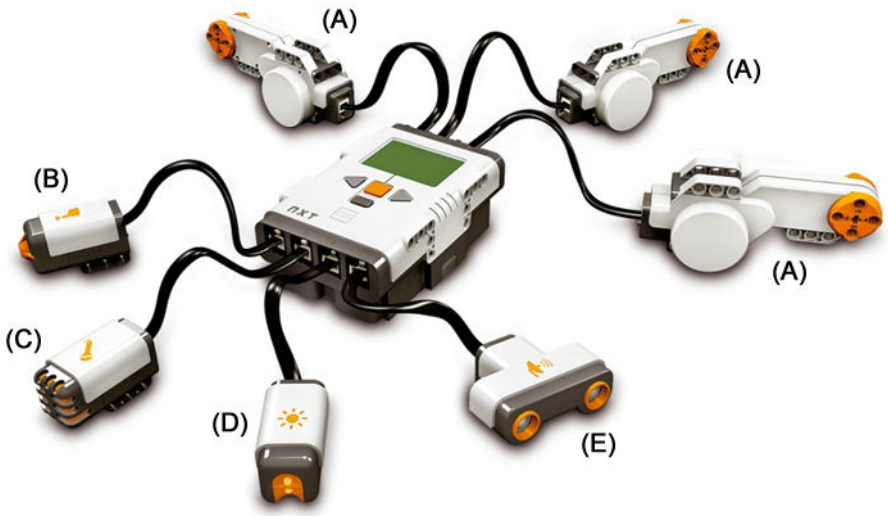


Şekil 3.3 Çalışma mantığının akış diyagramı

BÖLÜM DÖRT
"ÇOK SİLAHŞORLAR" YAZILIM PROJESİNİN İÇ YAPISI

4.1 Lego Mindstorms NXT

"Lego Mindstorms NXT", Lego firmasının 2006 yılında piyasa çıkardığı, programlanabilir robotik setidir. İlk nesil "Lego Mindstorms" kitinin yerini alan bu set NXT-G programlama yazılımı ile birlikte gelmesine rağmen, Lego'nun cihazın gömülü yazılımını açık kaynak olarak yayınlaması ile birçok yazılım dili için kütüphaneleri bulunmaktadır. Lego'nun yazılımcılar için sunduğu üç çeşit yazılım kiti bulunmaktadır. Lego SDK yardımıyla, cihazın sahip olduğu USB bağlantısından bilgi akışı sağlanabilmektedir. Lego HDK, cihazın ve kullandığı algılayıcıların detaylı bilgileri ve şemaları yer almaktadır. Lego BDK, "Bluetooth" bağlantısı hakkında detaylı bilgilere sahiptir (Lego Mindstorms NXT,b.t.).

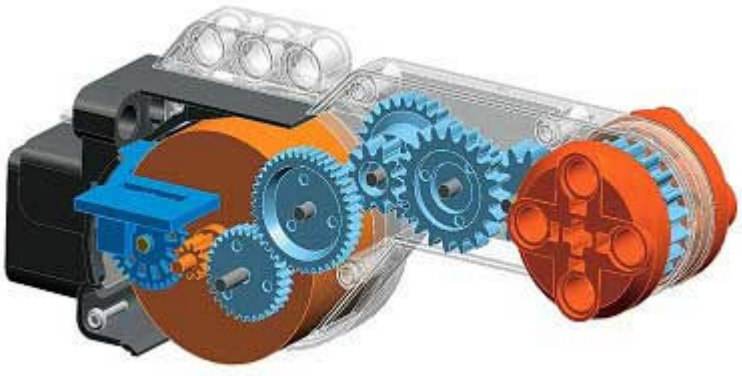


Şekil 4.1 "Lego Mindstorms NXT" kitinin ekipmanları: **(A)** Servo motorlar, **(B)** dokunma, **(C)** ses, **(D)** ışık ve **(E)** ultrason algılayıcısı (Lego,2006)

Düşük enerji tüketimli, standart (IEEE 802.15) bir iletişim protokolü "Bluetooth", RS232 veri kablolarının kablosuz alternatifi olarak geliştirilmiştir. Birden fazla sabit ve mobil cihazlar için kullanıma uygundur. Cihazlar arasındaki eşleme sorunlarına da çözüm getirmiştir. Günümüzde bu teknolojiyi kişisel ve sanayi alanında kullanımı

oldukça geniştir. Cihazlar radyo yayını kullandıkları için birbirlerinin görüş alanında olmaları gerekmemektedir. Radyo yayın menzili için 1, 10 ve 100 metre olmak üzere 3 farklı seviye bulunmaktadır. Genellikle 2. seviyeye sahip bir cihaz, 1. seviye bir cihaz ile bağlandığında menzilinin uzadığı tespit edilmiştir. "Bluetooth" kablosuz iletişim protokolünün ilk sürümü (1.2) 1 Mbit/s, ikinci sürümü (2.0) 3 Mbit/s'lik veri iletişim oranına sahiptir (Sallhammar,2004).

Bu setin, en temel bileşeni şekil 4.1'de görüntüsü verilen "NXT brick" adlı bilgisayarıdır. "NXT brick" 4 algılayıcıya (ultrason, dokunma, ışık ve ses) ve 3 servo motora (bkz. Şekil 4.2) dahili redüktörlü ve optik encoder'lı) kadar kontrol desteği vermektedir. Servo motorlar 1/48 oranlı redüktöre ve 10/32 oranlı, 12 delikli encoder diskine sahiptir. Motorun bir dönüşünde encoder diski $48 \times \left(\frac{10}{32}\right) = 15$ kez dönmektedir. Her iki tarafından delik olduğu için $15 \times (2 \times 12) = 360$ birim okumaktadır (ColoRotate,b.t.). Bu algılayıcılar ve motorlar RJ12 standardındaki kablolar ile bağlanabilmektedir. "NXT brick" üzerinde bulunan 100x64 piksellik LCD ve 4 adet buton yardımıyla içinde bulunan gömülü yazılım kontrol edilebilmektedir. Ayrıca içinde bulunan hoparlör yardımıyla 8 kHz örnekleme oranına kadar ses dosyalarını desteklemektedir. 6 adet AA cinsi 1,5 voltluk pil ile beslenen cihaz, ek olarak Li-on şarjlı pilleri ile de kullanılabilir. Cihazın kullanım esnasında daha hızlı pil değişikliği yapılabilmesi ve maliyetin azaltılabilmesi açısından, cihaz 9 voltluk pil ile çalışılabilecek konuma getirilmiştir.

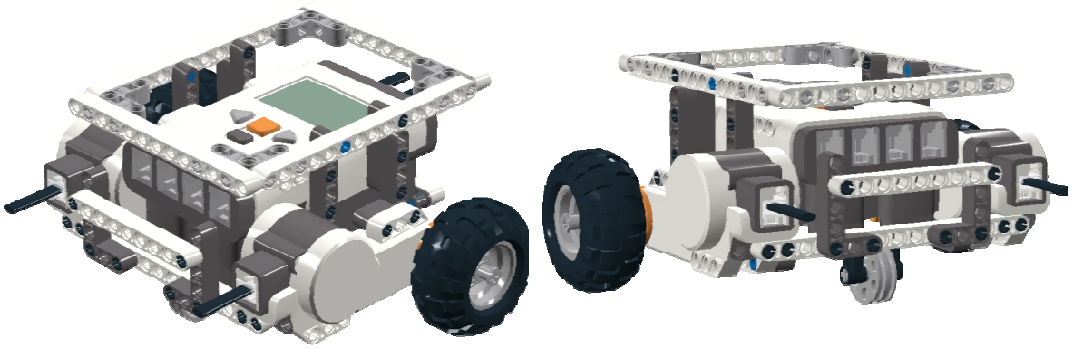


Şekil 4.2 "Lego Mindstorms NXT" kitinin servo motoru (Color Models,b.t.)

"NXT brick" cihazı 32 bit 48 MHz'lik ARM serisi mikroişlemciye, 256 KB flash hafızaya ve 64 KB RAM belleğe sahiptir. Bu mikroişlemci yaygın olarak elektronik cihazlarda (örneğin Apple iPod, Nokia, Nintendo DS, iRobot Roomba 500 serisi gibi) kullanılmaktadır. Mikrodenetleyici olarak, 8 bit 4 MHz'lik ATmega48 serisi mikrodenetleyiciye sahiptir. "Bluetooth" bağlantısının kapsama alanı yaklaşık 10 m'lik 2.seviyededir.

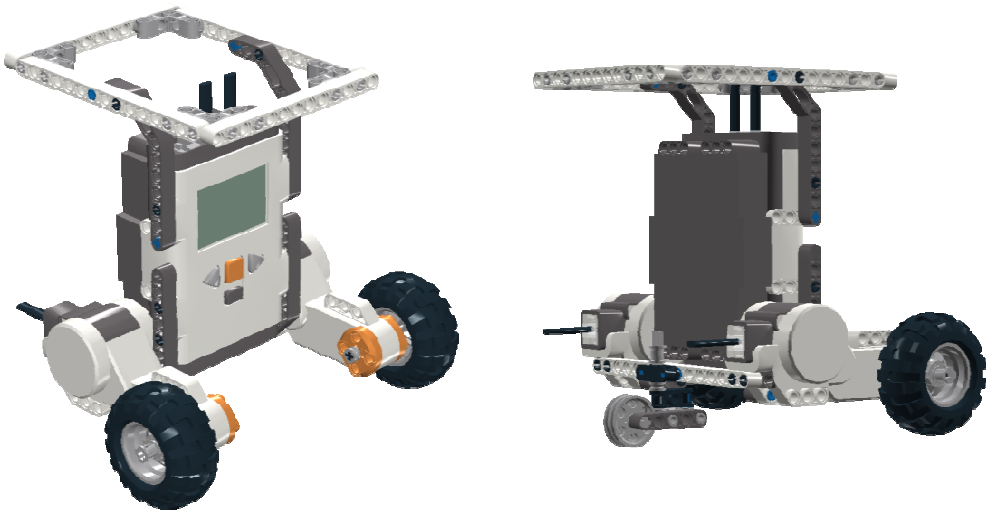
4.1.1 Mobil araç tasarımları

Proje geliştirilirken en uygun mobil araç tasarımı bulunması hedeflenmiştir. En uygun tasarımın az parça kullanması, farklı görevlere kolay uyum sağlayabilmesi, her hangi bir parçanın pil değişimini engellemeyecek olması, sürüş dengesini ani manevra ve hız değişimlerinde de sürdürebilmesi olarak belirlenmiştir. Manevra kolaylığı için tasarlanan ön dingilin fazla parça kullanması sebebiyle tasarımdan vazgeçilmiştir. Sürüşün her koşulda dengeli olabilmesi için düşünülen 4 tekerden çekiş, manevra yeteneğini olumsuz etkilediği için kullanılmamıştır. Bu denemeler sonucunda istenen koşullara en uygun tasarım, iki çekişli 3 tekerlekli bir araç tasarımına geçilmiştir. Bu çeşit tasarımlarda manevra yetenekleri istenen seviyede olsa da dengesini koruması konusunda sıkıntılar bulunmaktaydı.



Şekil 4.3 “Lego Digital Designer” yazılımı ile çizilen ilk araç

Denemeler sonucunda karar iki tasarım ile bu sorunlarda ortadan kaldırılmış, tüm isteklere cevap verebilen mobil araçlar tasarlanmıştır. Tasarımlar ayrıca ileride tekrarlanabilir olması için "Lego Digital Designer" programı ile sayısallaştırılmıştır. Mobil araçları tanımlayan renk etiketleri araçların üzerinde bulunan dikdörtgen çerçeveye yapıştırılmaktadır.



Şekil 4.4 "Lego Digital Designer" yazılımı ile çizilen ikinci araç

4.2 Kullanılan Diğer Yazılım Kütüphaneleri

Proje kapmasında bilgisayarlı görü işlemlerini ve "Lego Mindstorms NXT" cihazlarının kontrolünü kolaylaştırmak, hızlandırmak ve geçmiş bilgi birikimlerinden de yararlanılması amacıyla "AForge.NET" isimli yazılım kütüphanesi kullanılmıştır. "EmguCV" yazılım kütüphanesi, bilgisayarlı görü çalışmalarında sıklıkla kullanılan OpenCV yazılım kütüphanesinin ".NET çerçevesi" için hazırlanmış çevreleyicisidir (wrapper). "AForge.NET" ile bilgisayarlı görü işlemlerinin yapılmasına rağmen kameradan görüntü almak amacıyla fazladan "thread" kullanması sürekli bilgilerin güncellendiği arayüzde takılmalara ve programın yanıt vermemesine sebep olmaktaydı. Özellikle bu sorunu ortadan kaldırmak için "EmguCV" kullanılmış ve "thread" mantığı terk edilerek, "zamanlayıcı (timer)" mantığı benimsenmiştir. "EmguCV" kullanımının bir diğer faydası, "Çok Silahşorlar" projesinin "Windows" işletim sistemi bağımlılığını ortadan kaldırarak, "Linux" çekirdek tabanlı veya "Mac OS X" işletim sistemlerinde de çalışabilmesi öngörülmüştür (Canming,b.t.).

4.2.1 AForge.NET

"AForge.NET" yazılım çerçevesi, bilgisayarlı görü ve yapay zekâ geliştiricileri, araştırmacıları için tasarlanmıştır. Yazılım çerçevesinin sahip olduğu kütüphaneler ve özellikleri (Kirillov,b.t.):

- AForge.Imaging: Standart görüntü işleme ve filtreleme kütüphanesi
- AForge.Vision: Bilgisayarlı görü kütüphanesi
- AForge.Neuro: Yapay sinir ağları hesaplama kütüphanesi
- AForge.Genetic: Evrimsel programlama kütüphanesi
- AForge.MachineLearning: Makine öğrenme kütüphanesi
- AForge.Robotics: Birkaç robot kitlerinin desteklenmesi için hazırlanan kütüphane
- AForge.Video: Video işleme kütüphanesi

“Çok Silahşorlar” projesi, “AForge.NET Imaging” ve “AForge.NET Robotics” kütüphanelerinden faydalanmıştır.

4.2.2 EmguCV

"EmguCV" işletim sistemleri arası taşınabilir, “OpenCV”'nin “.NET” çerçevesi için hazırlanmış bir çevreleyicidir (wrapper). Bu sayede “OpenCV”'nin fonksiyonlarını “.NET” destekli C#, VB, VC++, IronPython gibi dillerle de çalışması sağlanmıştır. Çevreleyici “Mono” çerçevesi ile “Linux” çekirdeğine sahip ya da “Mac OS X” işletim sistemlerinde de çalışabilir.

“EmguCV”nin sahip olduğu temel katman “OpenCV”'nin tüm yapılara destek vermektedir. İkincil katman sayesinde, var olan özelliklerin yanına “.NET” çerçevesinin getirdiği ek özelliklerde eklenmiştir.

BÖLÜM BEŞ
"ÇOK SİLAHŞORLAR" YAZILIM PROJESİNİN UYGULAMASI

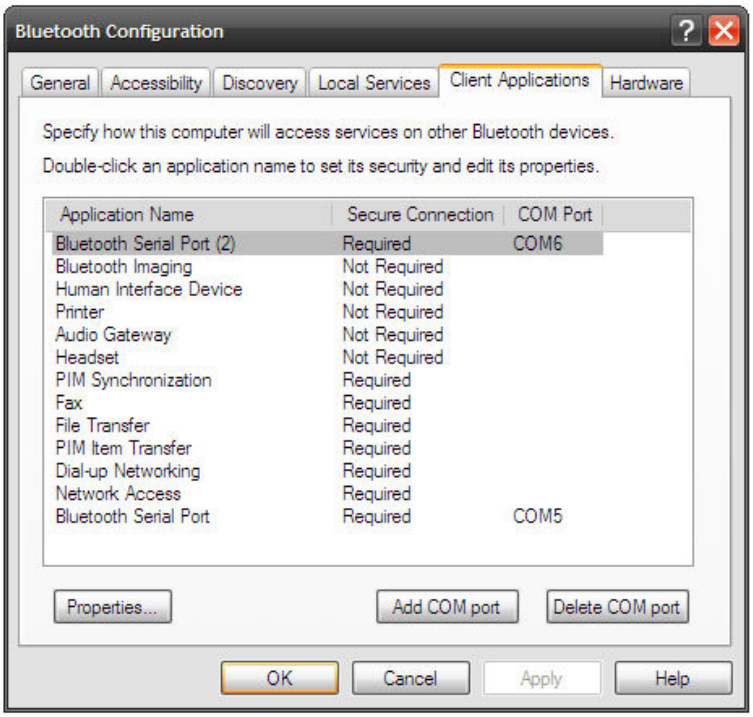
5.1 Tanıtım

Proje kapsamında mümkün oldukça basit fakat yazılım genelini kapsamlı biçimde kontrol edebilecek arayüz tasarımı yardımı ile çeşitli senaryolar planlanıp, gözlenebilmektedir. Senaryoların birleşimleri ile uygulamalar genişletilebilir. Örneğin mevcut ortamda bulunmayan, fakat belirli limitlerde hareket etmesini istediğiniz mobil araçlar için bilgisayar faresi yardımıyla sanal olarak engel tanımları yaparak, istenen limitleri tanımlanabilir. Eğer elimizde "Lego Mindstorms NXT" kiti bulunmuyor yada çalıştırılması uygun değil ise (pil sıkıntısı yaşanabilir) yazılım simülasyon amacı ile de kullanılabilir. Bu sayede mobil araçların hareketleri gözlenebilir.

5.2 Arayüzün Kullanımı

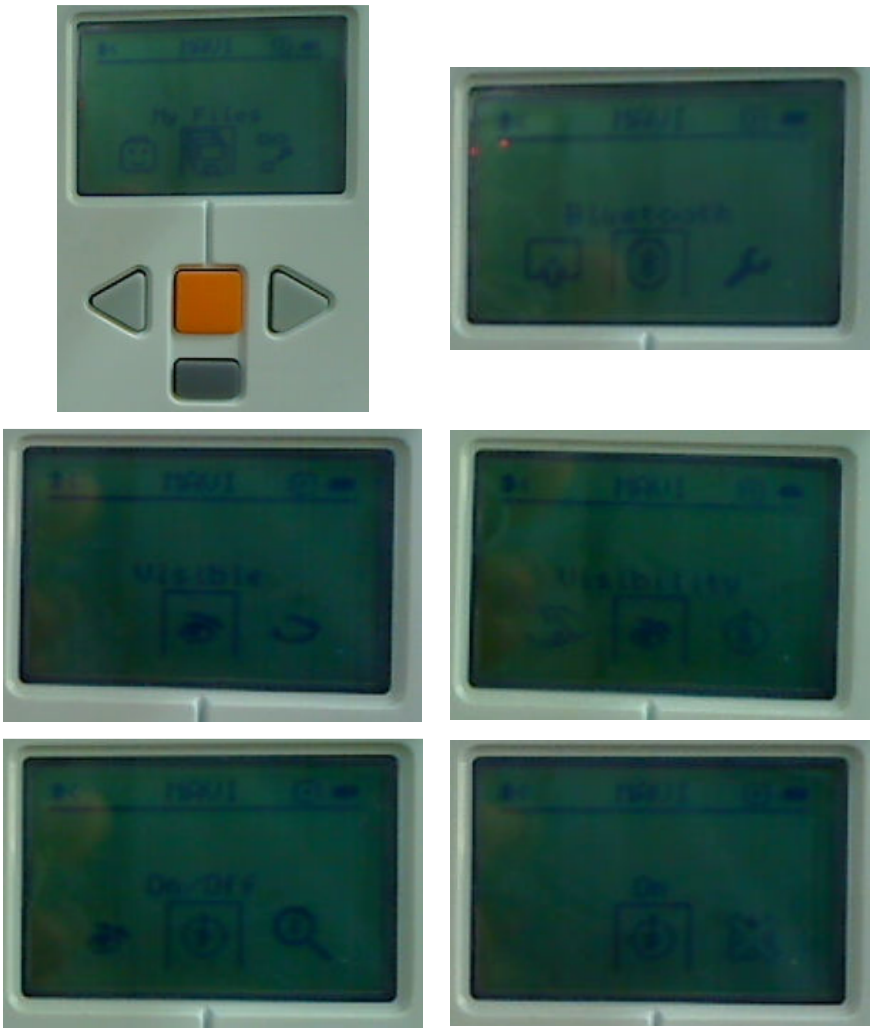
"Çok Silahşorlar" projesi ile birlikte geliştirilen arayüzün mobil araçlarla birlikte kullanımına başlamadan önce ilk olarak kullanılacak kontrolcü bilgisayarın dâhili ya da harici bluetooth donanımına sahip olması ve bu donanımın seri iletişim bağlantısını desteklemesi gerekmektedir.

Bluetooth donanımının gerekli ve yeterli özelliklere sahip olduğunu kontrol etmek için denetim masasından ya da başlat çubuğunun sağ tarafında bulunan ilgili simgeden ayar penceresi açılmalıdır. Firmaların donanımları ve buna bağlı olarak yazılımları farklılık göstereceği için özellikle bakılması gereken anahtar kelimeler "add" ya da "create" dir (bkz. Şekil 5.1). İlgili ayarların bulunmasından sonra kullanılacak mobil araç sayısına uygun bağlantı oluşturulur.



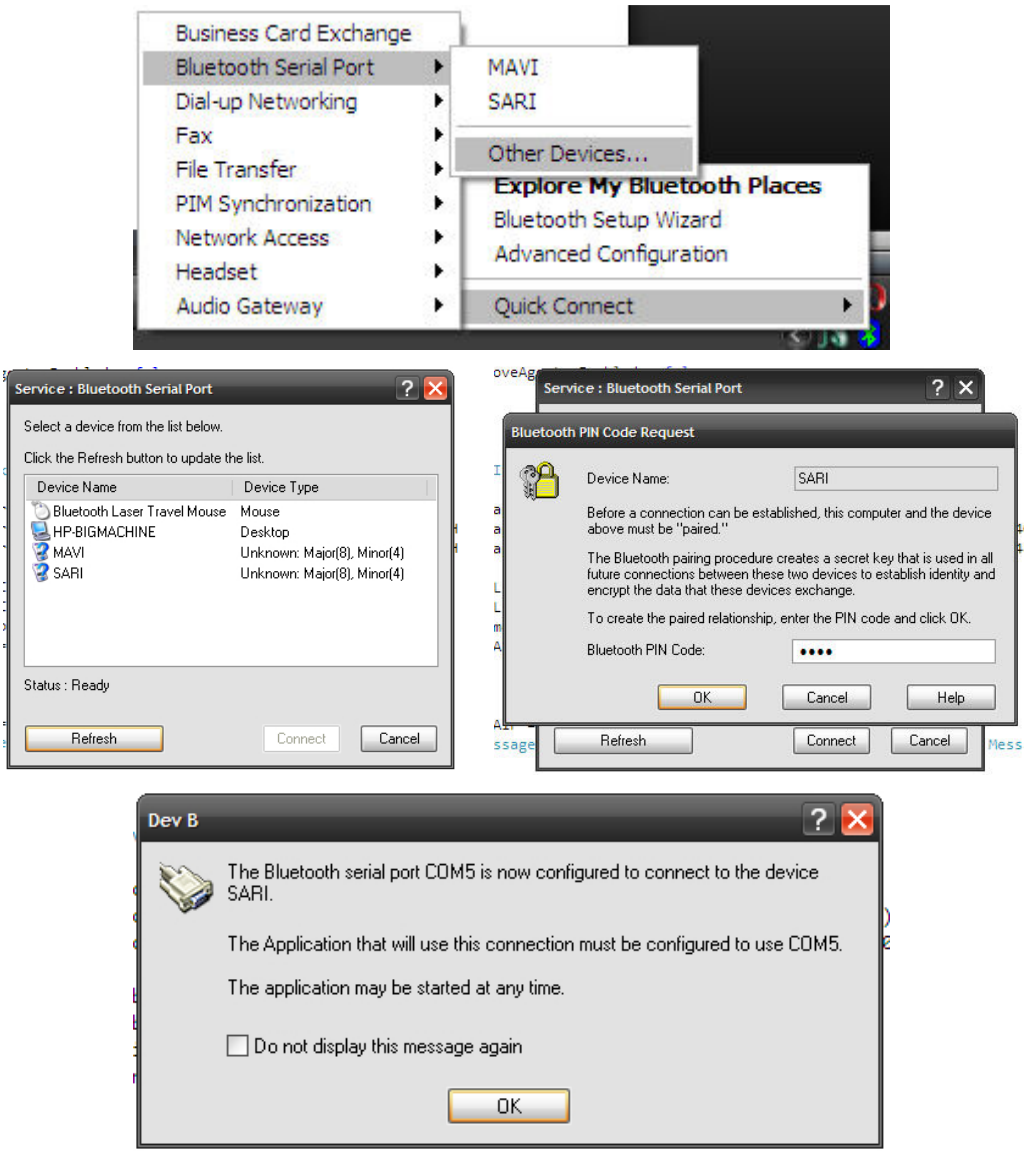
Şekil 5.1 Bluetooth seri iletişim bağlantı ayarlar penceresi

Seri iletişim bağlantıları başarı ile oluşturulduktan sonra araçları bir defaya mahsus olmak üzere kontrolcü bilgisayar ile eşleştirilmesi gerekmektedir. Eşleştirme işlemi için mobil araçlar bluetooth ile bağlantı kurabilecek duruma getirilmelidir. Bu ayarlar için öncelikle "Lego Mindstorms NXT" üzerindeki turuncu düğmeye basarak akıllı tuğla açılır. Yön tuşları yardımıyla bluetooth menüsüne ulaşılır ve yine turuncu düğme ile ayarlara girilir. "Visibility" (görülebilirlik) ayarının "visible" (görülebilir) olduğundan emin olunarak, "On/Off" ayarı "On" konumuna turuncu düğme yardımıyla bluetooth açık hale getirilir (bkz Şekil 5.2).



Şekil 5.2 Akıllı tuğlanın genel menüsü, kontrol düğmeleri ve bluetooth ayarları

Kontrolcü bilgisayar üzerinden seri iletişim bağlantısı yapacak cihaz aramaya başlanır. Bulunan mobil araç seçilerek, eşlemenin güvenliği için gerekli şifre adımına geçilir. Akıllı tuğlaların varsayılan şifresi “1234” dür. Bu şifre aynı zamanda akıllı tuğla üzerinde de görüntülenecektir. Akıllı tuğladakini onayladıktan sonra kontrolcü bilgisayara da aynı şifreyi girerek eşlemenin tamamlanmasını sağlanır. Eşleme adımı başarıyla tamamlandığında, seri iletişim bağlantı numarasını içeren bilgi ekranı karşınıza gelecektir (bkz Şekil 5.3).

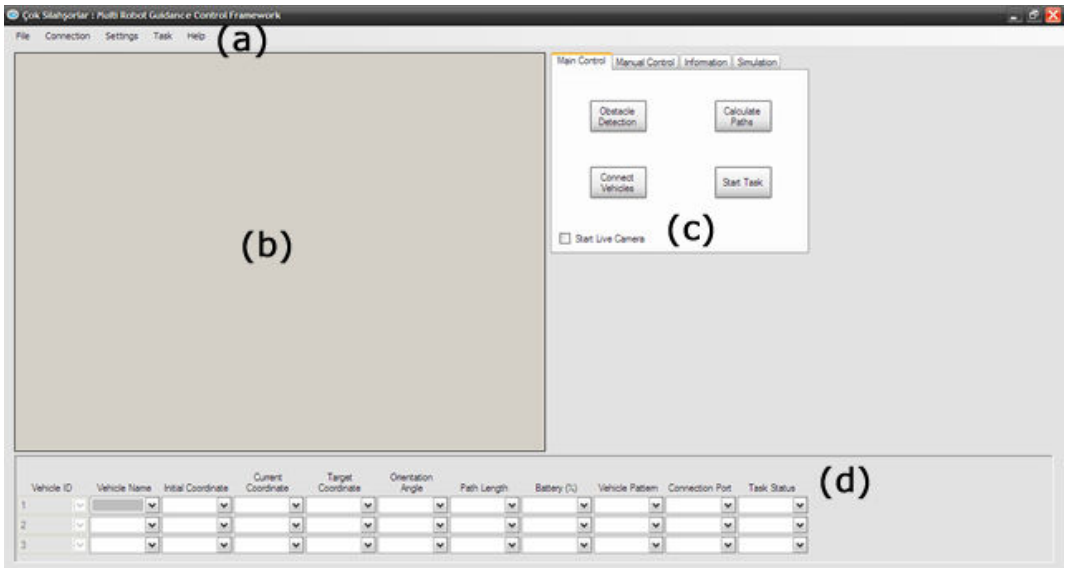


Şekil 5.3 Kontrolcü bilgisayar üzerinde bluetooth eşlemesi için yapılacak işlemler ve alınan sonuç

Tüm mobil araçlar için eşleme ve seri iletişim bağlantıları tahsis edildikten sonra "Çok Silahşorlar" programından sistemde mevcut seri iletişim bağlantıları kolaylıkla seçilip kullanılabilecektir.

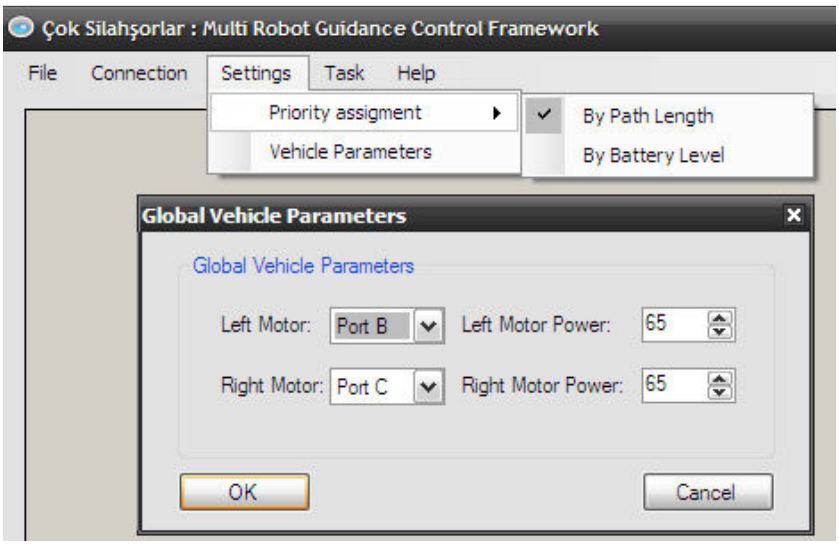
Programın arayüzü mümkün oldukça sadeleştirilmiş; en az düğme ve ayar ile sistemin çalıştırılması, kontrol edilmesi ve düzenlenmesi amaçlanmıştır. Arayüzü genel olarak; (a) klasik menü çubuğu, (b) canlı kamera görüntüsünün, araçların, engellerin, hesaplanan yolların gösterildiği kontrol, (c) görev ve bilgi için ayarların

yapıldığı sekmeli kontrol alanı ve (d) araçların tüm bilgilerinin tutulduğu tablo yapısından oluşmaktadır (bkz Şekil 5.4).



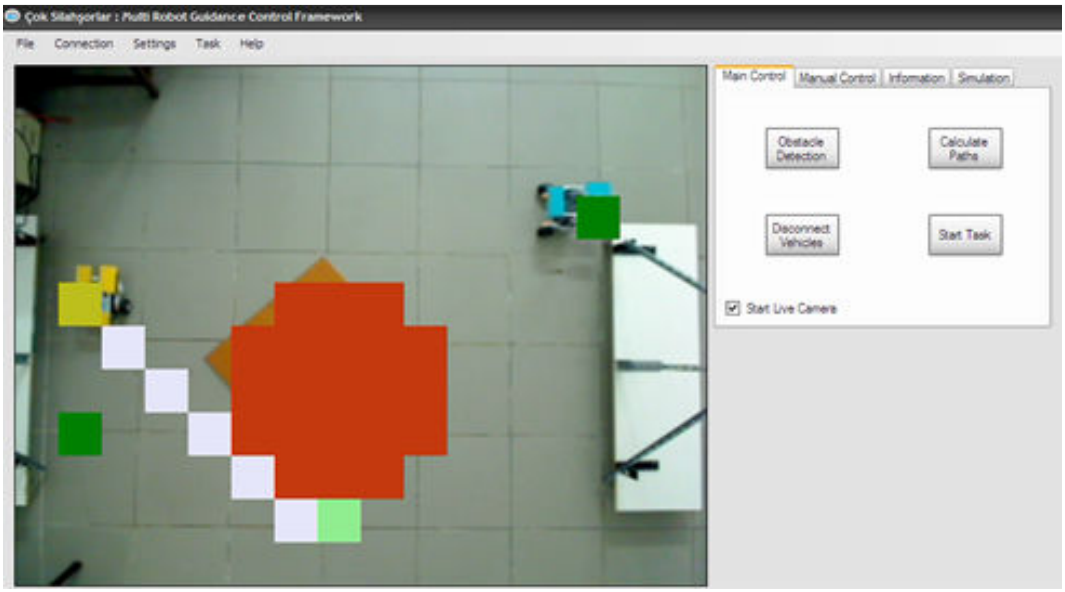
Şekil 5.4 "Çok Silahşolar" projesinin arayüzüne genel bakış; (a) önemli ve detaylı ayarların barındığı menü çubuğu, (b) durum ve yol bilgi ekranı (c) ana kontrol ve araç bilgi ekranı (d) araç ve görev takip tablosu

Arayüzün klasik menü çubuğu ana kontrol düğmelerini tetiklemenin yanı sıra bazı detay ayarlar pencerelerini de kullanıcıya sunmaktadır. Detay ayarları arasında öncelik sıralamasının tür (yol uzunluğu ya da pil seviyesi) seçimi ve mobil araçların motor bağlantı noktaları ile hız değerlerinin girildiği ayar penceresidir (bkz Şekil 5.5).



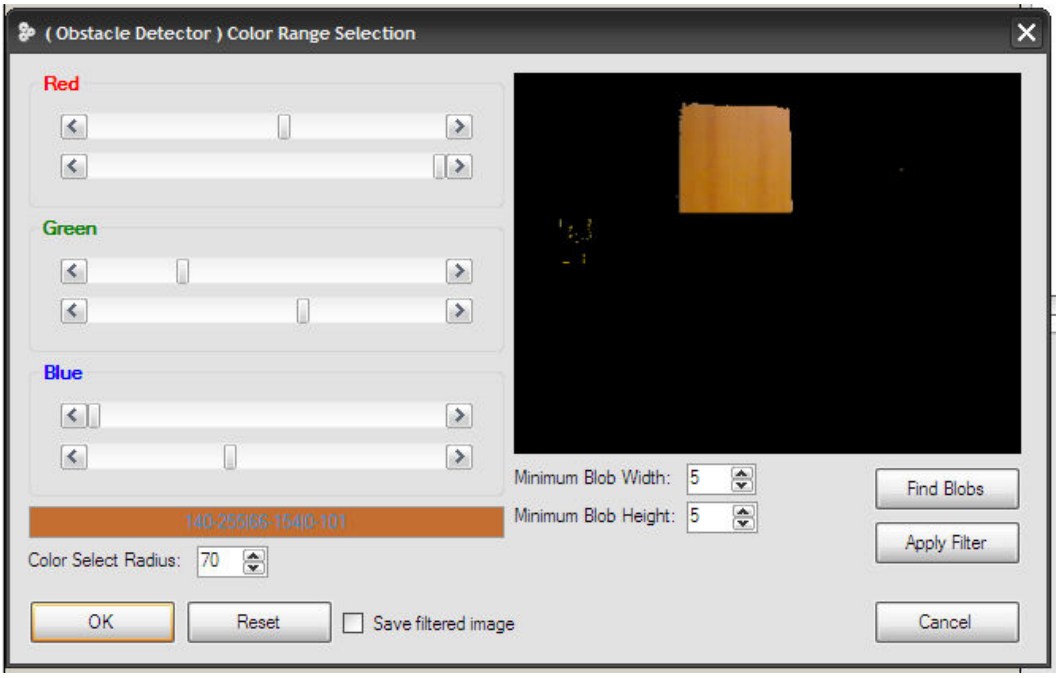
Şekil 5.5 Sadece klasik menü çubuğundan erişilebilen detaylı ayarlar

Arayüzün çoğunu kaplayan alan kullanıcı seçimiyle kameradan alınan görüntünün yansıtıldığı, aynı zamanda araçların ve engelin renklerine göre işaretlendiği, hesaplanan yolların çizildiği görselliğin ön plana çıktığı bir form elemanıdır (bkz Şekil 5.6). Bu form elemanı ".NET Framework" ün "PictureBox" form elemanın miras alarak ek özellikler eklenmiş, projeye özel bir form elemanıdır.



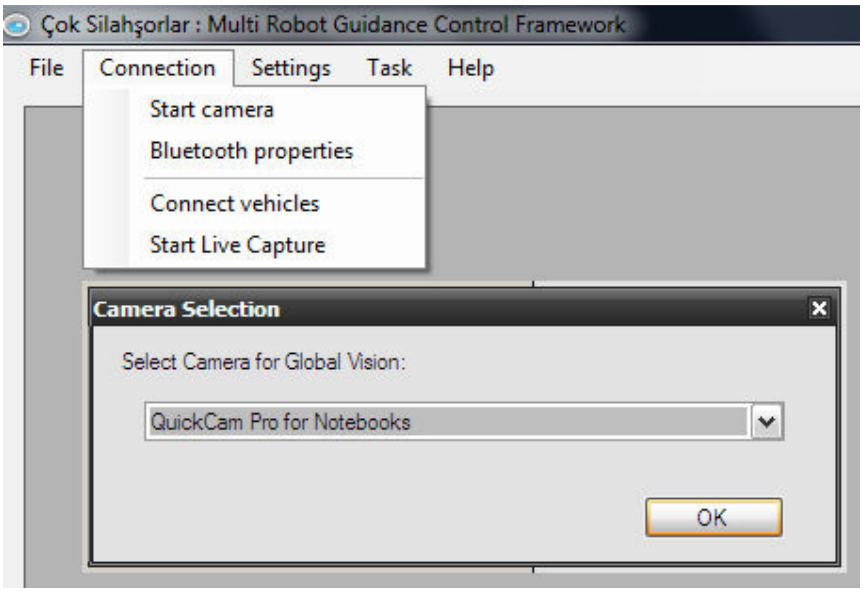
Şekil 5.6 Görselliğin ön plana çıktığı, tanımlama ve kontrol işlemlerini kolaylaştıran form elemanı kullanılırken. "Start Live Camera" seçeneği işaretlenerek kameradan canlı görüntünün gösterilmesi sağlanılmıştır.

Sekmeli kontrol alanının ilk sekmesi olan "Main Control" de 4 düğme ve bir işaretleme kutusu bulunmaktadır. Düğmelerden "Obstacle Detection" ile engel renk tanım yapılır. Açılan engel rengi tanımlama penceresi (bkz. Şekil 5.7) ister kaydırma çubuğu istenirse de farenin sol tıklaması ile istenen renk değeri seçilebilir. Seçilecek renk aralığını tanımlamak için "Color Select Radius" değeri düzenlenerek daha verimli sonuçlar alınabilir. Engel tanımlamasında kullanılan filtrelerin anlık sonucunu görmek için "Apply Filters" düğmesi kullanılabilir. "Save filtered image" işaretleme kutusu yardımıyla seçilen engelin resmi diske kaydedilebilir.



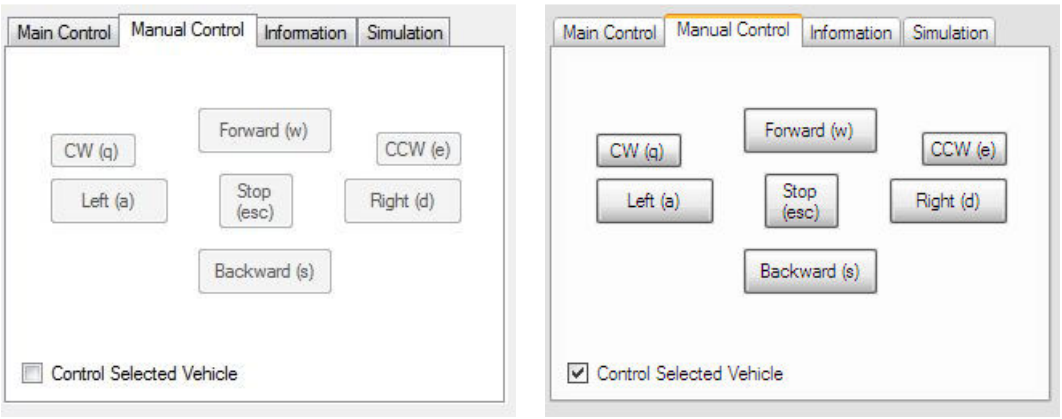
Şekil 5.7 Engel rengi tanımlama penceresi

Girilen ya da hesaplanan araç koordinatlarından hedef koordinatlarına olan en uygun yolu hesaplayan "Calculate Paths" düğmesidir. "Connect Vehicles" düğmesi yardımı ile seri iletişim bağlantı noktaları seçilmiş ve renk etiket değerleri girilmiş mobil araçlara bağlanıp, başlangıç noktalarını ve doğrultu açısını hesaplanır. Verilen görevi başlatan ise "Start Task" düğmelesidir. "Start Live Camera" işaretleme kutusunun işaretlenmesi durumunda canlı kamera görüntüsü arayüz üzerinden takip edilebilmektedir. Kamera gerektiren tüm adımlarda önceden her hangi bir kameraya bağlanılmadığı taktirde, kontrolcü bilgisayara bağlı kameraların bir listesi kullanıcıya sunularak, kullanılacak kameranın seçimi sağlanılmaktadır. Kullanılacak kameranın değiştirilmesi için klasik menü çubuğundan mevcut kameradan alınan görüntünün durdurulması ve tekrardan kamera seçimi yapılması gerekmektedir (bkz Şekil 5.8).



Şekil 5.8 Kamera bağlantısı başlatma ve durdurma menüsü ve bilgisayara bağlı kameraların seçim ekranı

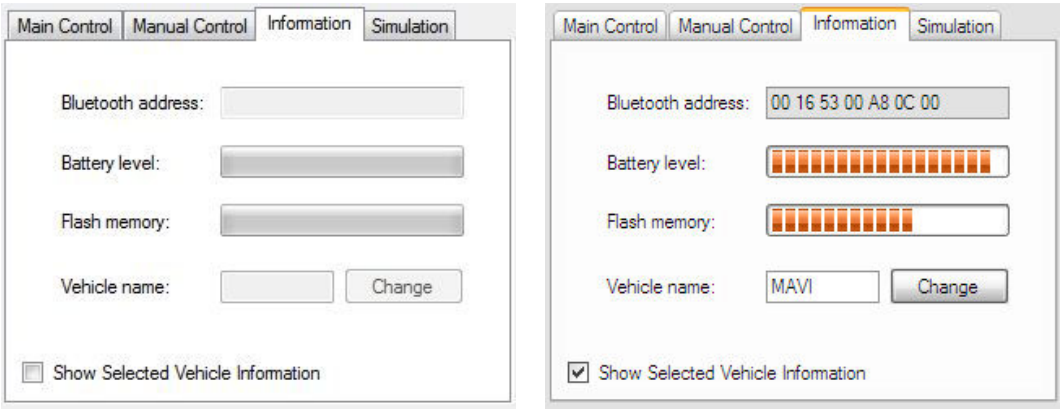
İkinci sekmede bulunan işaretleme kutusu yardımıyla tabloda seçilen satıra ait mobil aracın elle kumandaya geçirilmesi ve klavye ya da ilgili düğmeler yardımı ile kontrol edilebilmesi sağlanmıştır (bkz Şekil 5.9).



Şekil 5.9 Seçilen mobil aracın elle kumandaya geçirme ve hareket kontrolcüleri

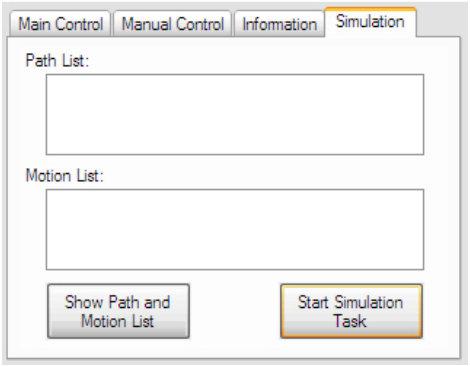
Sekmeli kontrol alanının üçüncü sekmesi seçilen mobil araç hakkında detaylı bilgileri vermekle birlikte kullanıcıya mobil aracı dilediği gibi isimlendirmesi için de ayrı bir seçenek sunar (bkz Şekil 5.10). Mobil araçların isimlerinin sahip olduklar renk etiketlerine göre isimlendirilmesi mobil araç sayısının artması durumunda karışıklıkları önleyen hatırlatıcı bir bilgi olarak yardımcı olur. Bu bilgilerin alınması ve isminin değiştirilmesi için mobil aracın öncelikle başarı ile bağlanması

gerekmektedir. Bağlantıda bir sorun olması halinde işaretleme kutusu seçime izin vermeyecektir.



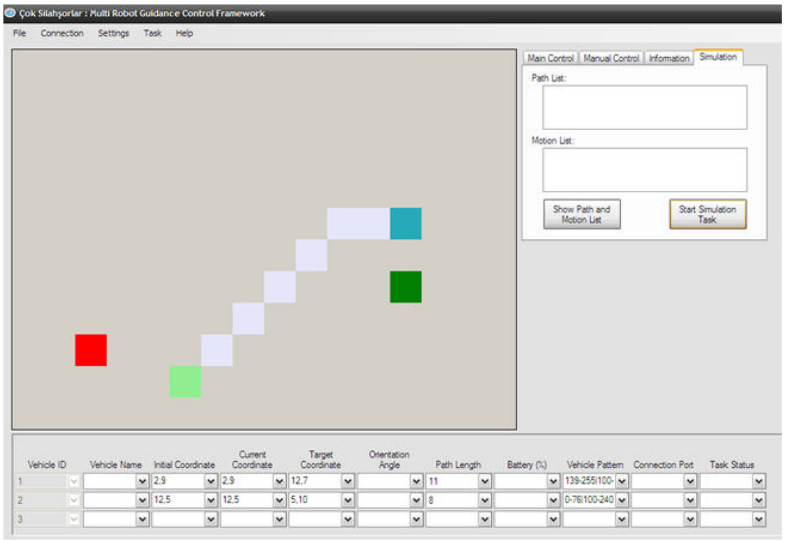
Şekil 5.10 Seçilen mobil aracın detaylı bilgileri (Bluetooth alıcı-verici adresi, pil seviyesi, hafıza seviyesi) ve isim değiştirme seçeneğini

Son sekmede mobil araçların istenildiği gibi tamamen gerçeğine uygun bir biçimde simülasyonunun yapılabileceği "Start Simulation Task" düğmesi ve seçilen mobil aracın hesaplanan yol ve hareket bilgilerinin adım adım listeleyecek "Show Path and Motion List" düğmesi bulunmaktadır (bkz Şekil 5.11). Simülasyon, özellikle teknik bir sebepten dolayı (örneğin aracın ya da araçların pillerinin az ya da tamamen bitmesi durumu) ya da mobil araçlara görevi doğrudan vererek zarar gelmesini engellemek amacı ile yapılabilir. Simülasyon için istenirse (bağlı kameranın bulunması durumunda) engel tanımı ilgili düğme ile yapılabileceği gibi canlı görüntünün yayınlandığı bölüme fare ile sağ tıklayarak da engel çizilebilir. Engel tanımı tamamen isteğe bağlıdır.



Şekil 5.11 Simülasyon görevinin başlatılması, yol ve hareket bilgilerinin listelenmesi için kullanılan düğme

Mobil araçların renk etiketleri normal görevlerde olduğu gibi seçilebileceği gibi önceden tanımlı olan renklerde seçilebilir. Simülasyon işlemi için gerekli olan kritik bilgiler: başlangıç ve ona eşit olan (ilk konumu olduğu için) anlık koordinat, hedef koordinatı, renk etiket değeridir (bkz Şekil 5.12). Hedef koordinat normal görevlerde olduğu gibi canlı görüntü alanına sol fare tıklaması ile seçilebilir. Elle yazılacak koordinat değeri için "x,y" formatının kullanılması gereklidir. Mobil araç bilgilerinin de girilmesinden sonra yollar hesaplatılır. Başarıyla yolların hesaplanması ile simülasyon işlemi başlatılabilir.



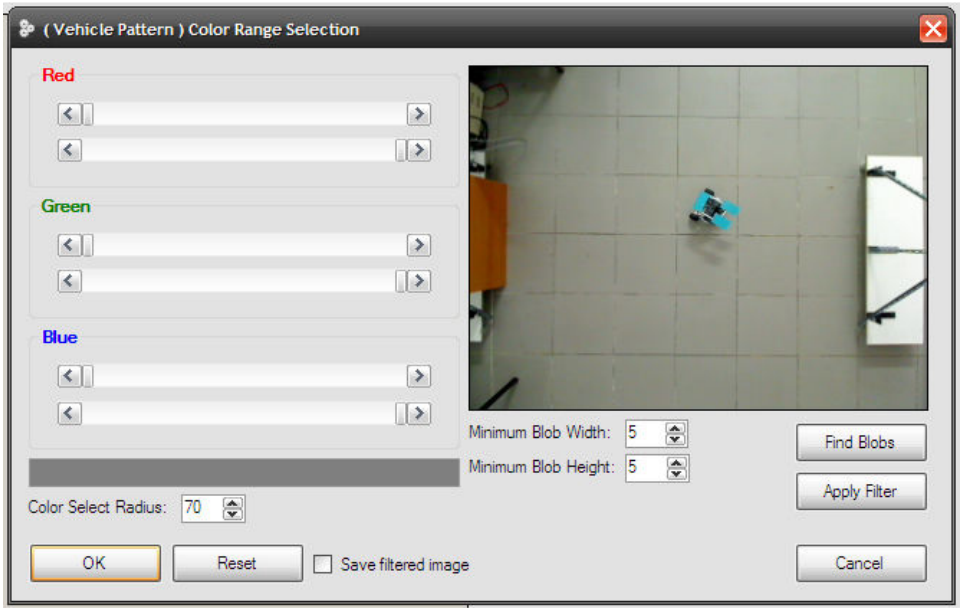
Şekil 5.12 Simülasyon görevi için gerekli ve yeterli bilgiler

Araç ve görev takip tablosu mobil araçlar hakkında tüm bilgilerin tutulduğu tek yerdir. Bu bilgiler araçların tekil numaraları, isimleri, başlangıç, anlık ve hedef koordinatları, doğrultu açısı, yol uzunluğu, pil seviyesi, renk etiketi, seri iletişim bağlantı noktası ve görev durumudur.

Tablo üzerinde yer alan "COM Port" sütunundan mobil araçların kontrolcü bilgisayar ile eşleşmesi sonucunda belirlenen seri iletişim bağlantı noktaları listelenmiştir. Bu seçim yapıldıktan sonra araç renk etiketinin belirlenmesine geçilebilir.

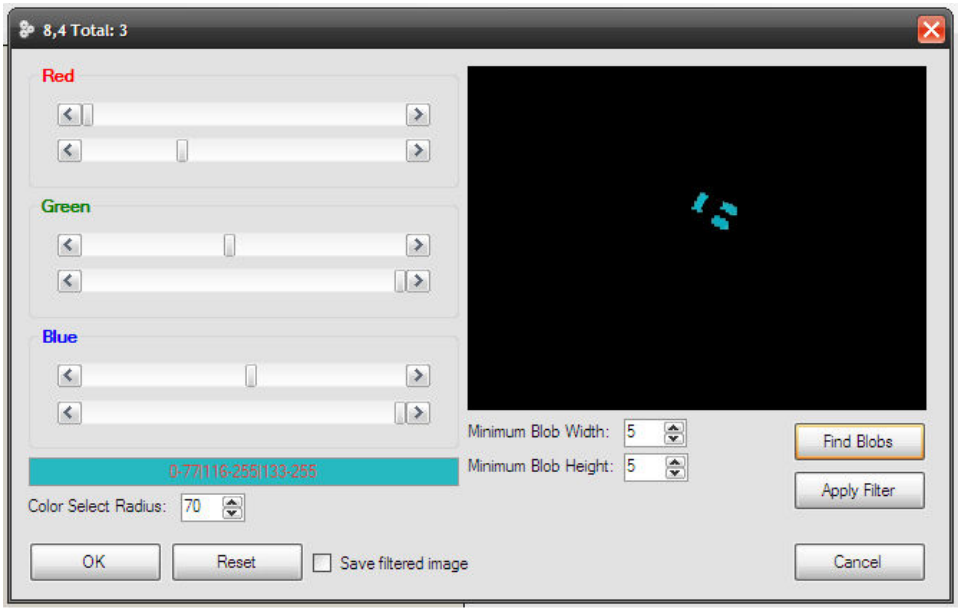
Mobil araçlara kamera yardımı ile renk etiketi belirlemek için ilgili kısımdan "Select .." seçilerek renk etiket belirleme penceresi açılabilir (bkz Şekil 5.13). Açılan pencereden isteğe bağlı olarak ya kaydırma çubukları yardımıyla ya da önizleme

görüntüsü üzerinden kullanılması istenen rengin üzerine farenin solu ile tıklanarak ilgili renk seçilip, ince ayarları yine kaydırma çubukları ile yapılabilir (bkz Şekil 5.14).



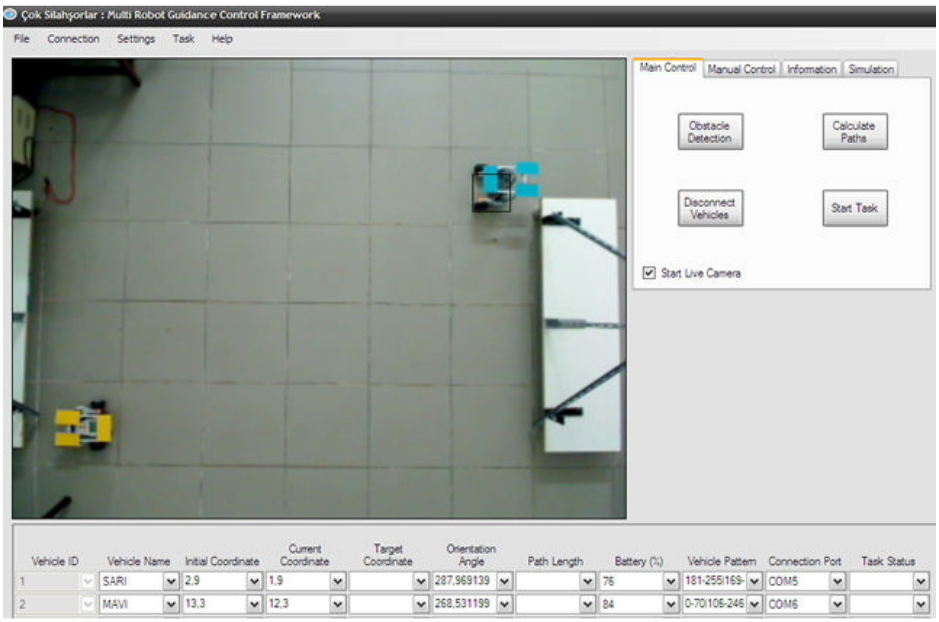
Şekil 5.13 Mobil araç etiket tanımlama ekran açılışı

Fare ile seçilen rengin en küçük ve en büyük renk aralığı ayarlanarak tıklama sonucunun daha verimli olması sağlanabilir. Bu ayar penceresinde ayrıca kabarcık analizi için gerekli olan en küçük kabarcık boyutlarının seçimi de yapılabilir. Yapılan kabarcık ayarlarının sonuçlarını anlık görmek için "Find Blobs" düğmesi kullanılabilir. Araç renk etiketi tanımı yapılırken kabarcık sayısının üç olması gerektiği için mutlaka bir defa kontrol etmek tavsiye edilir.



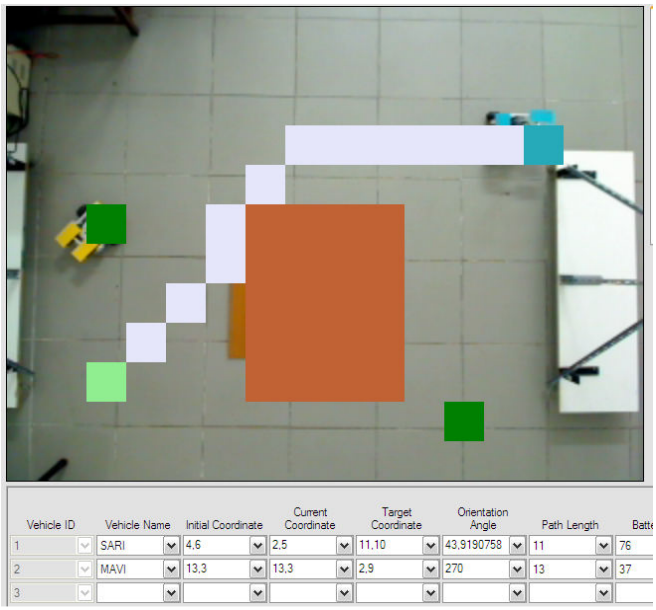
Şekil 5.14 Mobil aracın etiketi fare ile seçildikten sonra filtrelenmiş görüntüsü ve kabarcık analiz sonucunun pencereye yazdırılması

Tablo üzerinde bağlantı kurulacak her mobil aracın ilgili bağlantı noktası ve renk etiketi belirlendikten sonra araçlara bağlantı sağlanılabilir. Başarı ile bağlanıldıktan sonra girilen bilgiler doğrultusunda araç isimleri, başlangıç ve anlık koordinatları (İlk anda hareket ettirilmediklerinde birbirlerine eşittirler.), doğrultu açıları ve pil durumları tabloya eklenecektir ve her hangi bir değişim anında güncellenecektir (bkz Şekil 5.15).



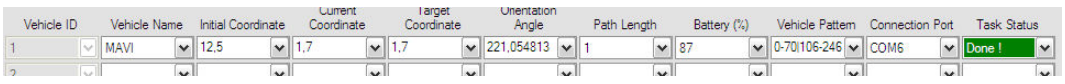
Şekil 5.15 Mobil araçların renk etiketleri tanımlanıp, bağlantı noktaları seçildikten sonra başarılı bir bağlantı sonucu

Mobil araçların bilgileri tabloya başarıyla işlendikten sonra araçlar tek tek seçilerek hedef koordinatları tanımlanabilir. Hedef koordinatları tanımlamak için kullanıcı ister canlı kamera görüntüsü desteğini kullanarak isterse de kullanmadan seçili araca elle hedef koordinat verebilir. Canlı kamera görüntüsü üzerinden farenin solu ile tıklanarak istenen (engel tanımlaması yapıldığı alanlar hariç) koordinat, seçilen araca hedef koordinat olarak tanımlanacaktır (bkz Şekil 5.16).



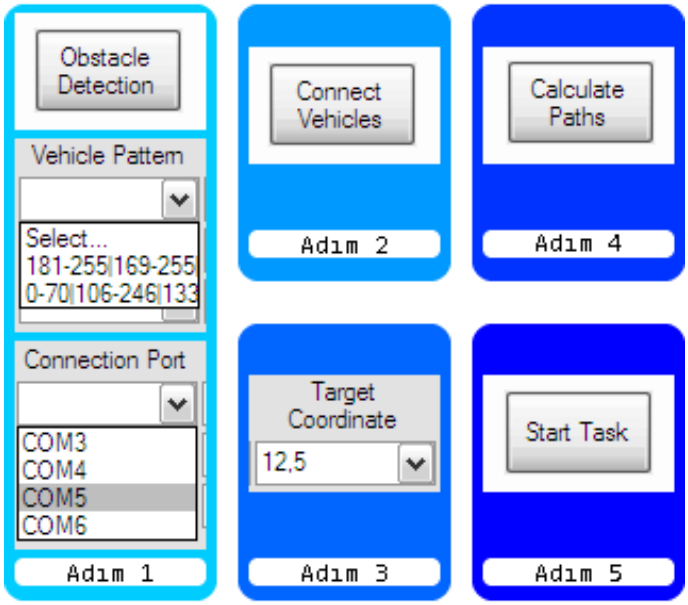
Şekil 5.16 Mobil araçlara hedef koordinatlarının verilmesi ve yollarının hesaplatılması

Hedef koordinatında girilmesinden sonra araçların gideceği yollar hesaplatılır. Hesaplanan yolların uzunluğu tabloya eklenir ya da güncellenir. Bu işlemden sonra araçlar görevlerine başlamak için hazırlardır. Görevlerini başarı ile bitiren araçların görev bilgi sütununda kullanıcıya gösterilir (bkz Şekil 5.17).



Şekil 5.17 Tüm değerleri girilmiş mobil aracın başarıyla görevini sonlandırması

Tüm bu adımların sonucunda "Çok Silahşorlar" programı kullanılarak araçlara birçok farklı senaryolar uygulanabilir.



Şekil 5.18 "Çok Silahşorlar" programının başarıyla çalıştırılması için uygulanması gereken temel adımlar

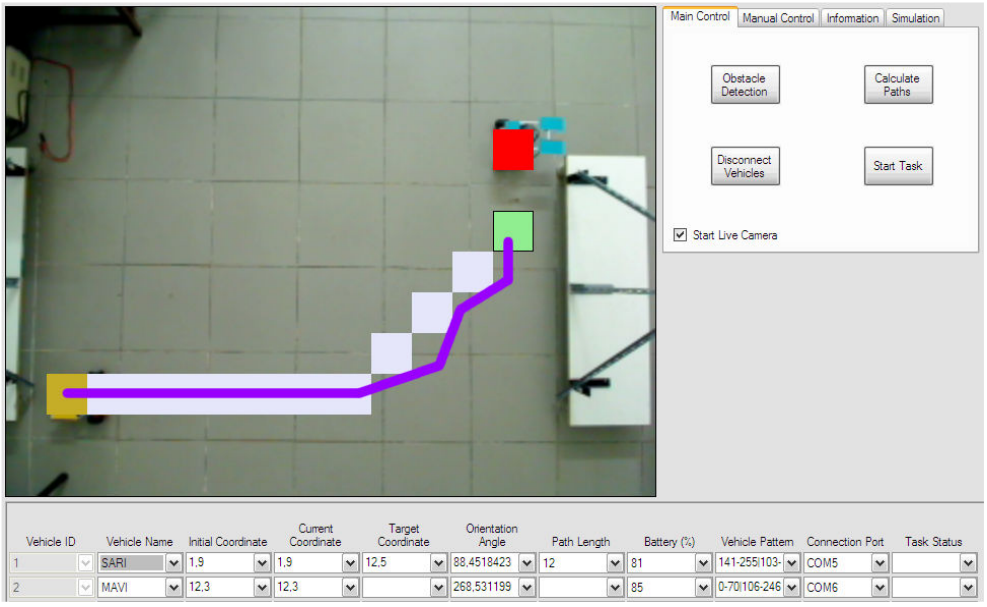
5.3 Verilerin Değerlendirmesi

"Çok Silahşorlar" projesi kapsamındaki hedefleri değerlendirmek üzere bir takım deneme senaryoları yapılmıştır. Bu senaryolar, bir önceki bölümde programın kullanımının anlatıldığı gibi ve iki farklı tasarıma sahip mobil araçla birlikte gerçekleştirilmiştir.

5.3.1 Hedefe yönelik görev

Senaryolardan ilki, genel görü sisteminin görüntü alanı içinde bulunan mobil araçların ayrı ve aynı zamanlarda kullanıcı tarafından belirlenen konumlara gitmeleridir. Bu senaryo, projenin en temel amacını oluşturmaktadır.

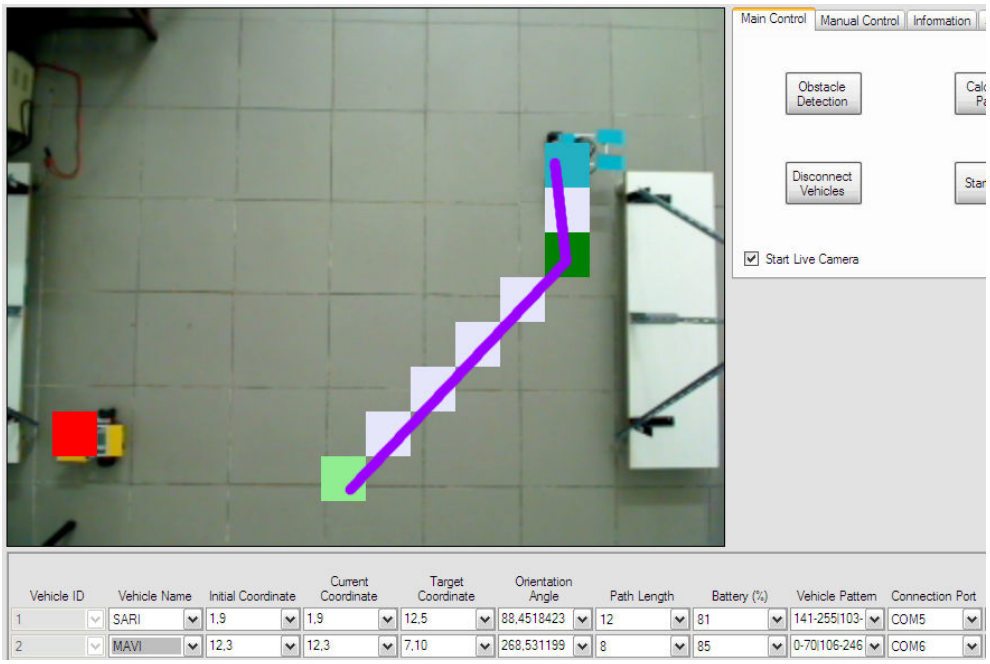
Kontrolcü bilgisayar, görüntü alanından geri besleme alacak kamera, mobil araçlar ve program hazırlandıktan sonra sarı renk etiketli mobil araca şekil 5.19'da görüldüğü üzere bir hedef noktası verilmiştir. Bu bilgiler yardımıyla hesaplanan yol kullanıcıya gösterilmiştir. Bu yolun beğenilmemesi ya da değiştirilmesi istendiği takdirde farenin sağ düğmesi ile sanal olarak engeller oluşturulabilir. Bu sanal engelleri de hesaba katan algoritma, yeni bir yol hesaplayacaktır.



Şekil 5.19 Tek bir mobil araca verilen göreve ait ekran görüntüsü; sarı aracın konumunu, yeşil hedef noktayı, bej renk hesaplanan yolu, mor ise mobil aracın gerçekte aldığı yolu göstermiştir.

Şekil 5.19'da görüldüğü üzere hesaplanan yol ile gerçekte mobil aracın aldığı yolda belirli bir hata bulunmaktadır. Bu hatanın en büyük sebebi sarı etiketli mobil aracın tasarımından kaynaklanmaktadır. Mobil aracın üçüncü tekerleği, araca sabitlenmiştir. Bu sebeple yapılan manevralara tam olarak uyum sağlayamamaktadır ya da manevrayı tam anlamıyla yerine getirememektedir. Bu olumsuz tasarıma rağmen, mobil aracın verilen hedef noktaya gitmesinin sebebi, her adımda tekrardan yolun hesaplanmasıdır. Her seferinde hesaplanan yolun sistemin büyümesi durumunda işleyişe zarar verme olasılığı bulunmaktadır. Bu olumsuz etkinin giderilmesi için yol hesaplamaları için kullanılan noktaların sayısı azaltılmıştır. Bu noktalar azaltılırken, mobil araçların boyutları da dikkate alındığından, sistemin hassasiyetinden ödün verilmemiştir.

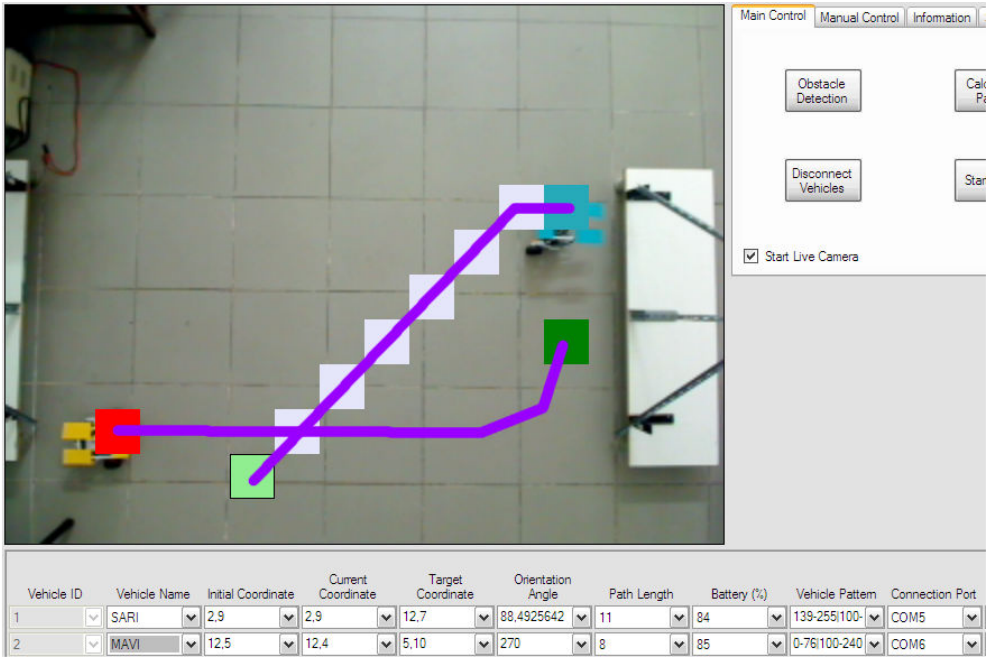
Aynı senaryoyu, tasarımı farklı olan mavi etiketli araca uyguladığımızda hesaplanan yol ile gerçekte gidilen yolun arasındaki hatanın azaldığı gözlenmiştir (bkz Şekil 5.20).



Şekil 5.20 Mavi etiketli mobil araca verilen göreve ait ekran görüntüsü; mavi aracın konumunu, yeşil hedef noktayı, bej renk hesaplanan yolu, mor ise mobil aracın gerçekte aldığı yolu göstermiştir.

Mavi etiketli aracın tasarımının, sarı etiketli araca göre en büyük farkı, üçüncü tekerleğinin tamamen serbest olmasıdır. Bu şekilde, verilen manevra komutlarına uygun cevaplar vermektedir. Mavi aracın başlangıçta gösterdiği kayma, aracın doğrultusunu düzeltmek için yapılan hamle sonucu meydana gelmiştir. Benzer kaymalar, kameradan saniyede alınan görüntünün arttırılması ile azaltılabilir.

Senaryonun son adımın, iki aracın aynı anda farklı hedef noktalara gitmeleri sağlanmıştır. Şekil 5.19'da, 5.20'de ve 5.21'de görüldüğü üzere sarı etiketli aracın konumu kırmızı renk, hedef noktası ise koyu yeşille işaretlenmiştir. Bu işaretlemenin amacı, mobil araçların birbirlerini engel olarak görmeleri ve yol hesaplamalarında birbirlerini de dikkate almaları gerektiğini belirtmektir.



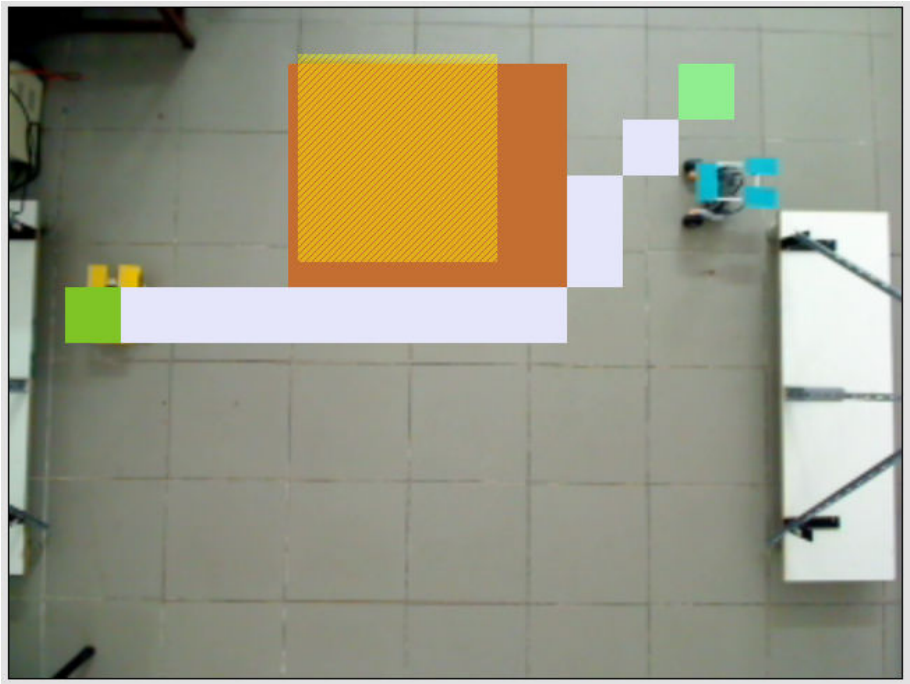
Şekil 5.21 İki mobil aracın aynı anda farklı hedef noktalara gitmeleri

Öncelik seçimi doğrultusunda ilk sıraya sahip araç görev boyunca her zaman ilk hareketi yapacaktır. Öncelikli araç hareket ettikten sonra diğer araçlar hareket edebilecektir. Bu sıralama sayesinde mobil araçların yollarının kesişmesi durumunda öncelikli araç yoluna devam eder. Yeni konumu, diğer araçlar için engel olarak tanımlandığı için diğer mobil araçlarda kaçış manevrası yaparak çarpışmadan yollarına devam etmektedirler.

5.3.2 Sabit engelli görev

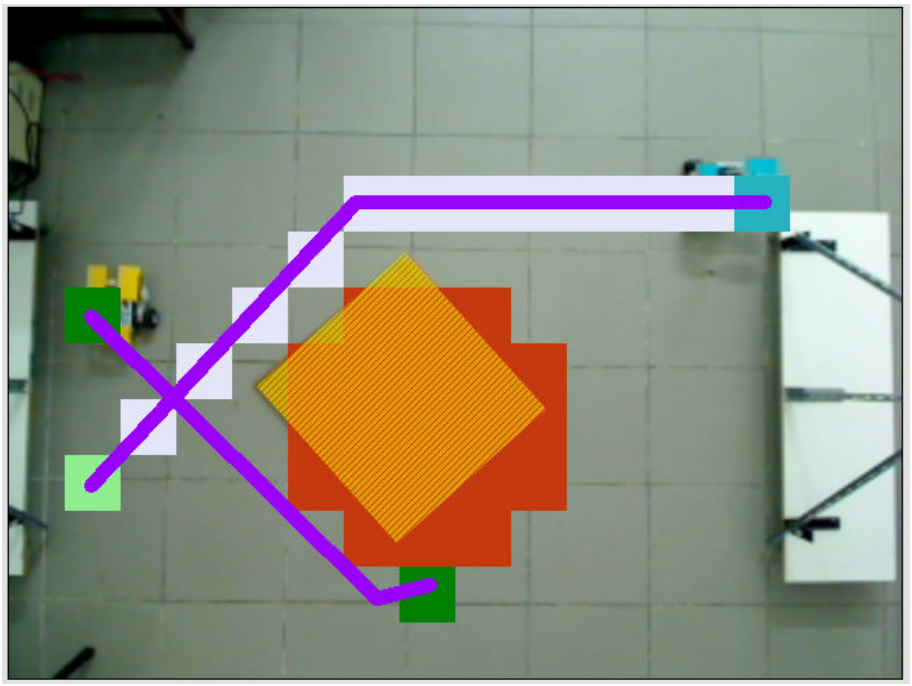
İlk senaryoda, mobil araçlar için görüntü alanı içinde herhangi bir engel tanımı yapılmamıştır. Tek engel, mobil araçların birbirleridir. Fakat görüntü alanı içinde var olan ya da sanal olarak eklenebilen engellere karşı da çözüm üretip, verilen görevlerin tamamlanması şarttır. Bu senaryonun amacı da görüntü alanı içinde tanımlanmış ya da farenin sağ tıklaması ile sanal olarak oluşturulmuş engelleri de hesaplanan yollara dâhil edilip, her hangi bir çarpışmanın engellenmesidir.

Görüntü alanı içinde bulunan sabit engelin tanımı, aynı mobil araçların renk etiketlerinin seçimi gibi kolaylıkla arayüz üzerinden yapılmaktadır. Engel tanımlandıktan sonra mobil araçlara gitmeleri istenen hedef noktalara yönlendirilir.



Şekil 5.22 Sabit engel tanımlandıktan sonraki ekran görüntüsü (Sarı ile taranmış alan engelin gerçek boyutlarını gösterir)

Engeller, mobil araçların büyüklüğündeki noktalar kümesinden oluşmak zorundadır. Bu zorunluluk, önceki bölümlerde tanımlanan çizge kuramında yer alan yol bulma algoritmalarının kullanımından kaynaklanmaktadır. Bu zorunluluğun avantajı, mobil araçların yolları hesaplanırken engelle arasında belirli bir mesafenin korunmasıdır. Mobil araçlar engellere teğet geçecekmiş gibi görüntülense de gerçekte güvenli mesafeyi her zaman korumaktadır. Dezavantajı ise engellerin düzgün olmayan ya da farklı bir açıda bulunması durumunda yol hesaplamaları için eksik bilgi verebilmeleridir.



Şekil 5.23 Belirli bir açı ile yerleştirilmiş engel tanımlandıktan sonraki ekran görüntüsü (Sarı ile taranmış alan engelin gerçek boyutlarını gösterir)

Şekil 5.23'de, şekil 5.22'deki aynı engel belirli bir açı ile görüntü alanına yerleştirilmiş ve sabit engel olarak tanımlanmıştır. Bu şekildeki tanımlama eksiklikleri ile karşılaşıldığında, canlı kamera görüntüsü yardımıyla tanımlanan sabit engel, sanal engel tanımlama ile en uygun hale kolaylıkla getirilir.

5.3.3 Hareketli engel görevi

Hareketli engel görev senaryosu, önceki senaryolar içinde, her aracın birbirlerini engel olarak görmeleri sayesinde, yer almaktadır. Araçlar sabit engeli, yol bulma ve planlama hesaplarına kattıkları gibi, hareketli engelleri de değerlendirirler. Her hareketten sonra yolların hesaplanmasının bir avantajı da cevre koşullarının değişmesi durumunda mobil araçların bunlara uyum sağlayabilmeleridir.

BÖLÜM ALTI
SONUÇLAR

6.1 Genel Açıklama

Görüntü alanı içinde hareket kontrolü konusundan yola çıkan "Çok Silahşorlar" projesi, çok ajanlı sistemlerin özelliklerinden ve faydalarından yararlanan, çok robotlu sistemlerin kontrolünü ve simülasyonunu mümkün kılan, kullanıcı dostu bir yazılım çözümü sunmuştur.

"Çok Silahşorlar" projesi, kendi sahip olduğu arayüzün yeteneklerini bir adım öteye götürerek, benzer çalışmalara yardımcı olmuştur. Bu yeteneklerin başında; simülasyon seçeneği, tek tık ile renk etiket seçimi, tek renk etiketi ile konum ve doğrultu hesaplama, yol uzunluğu ya da pil seviyesine göre öncelik verme gelmektedir.

Proje genelinde temel alınan modülerlik ilkesi sayesinde oraya çıkan alt programlar, tek başlarına dahi çeşitli çalışma konularına yardımcı olabilecek kapasiteye sahiptir. Örneğin "çoklu renk takibi" alt programı, insan yürüme analizi için kullanılabilmektedir.

6.2 İlerideki Çalışmalar

“Çok Silahşorlar” projesi hakkında ileride çalışılması düşünülen konuların başında; projeyi, arayüzü kolaylığında kullanılabilecek bir yazılım kütüphanesi haline getirmektir. Bu sayede benzer konularda çok daha esnek biçimde çalışılabilmesi sağlanacaktır. Bu esneklik, “Lego Mindstorms NXT” ile birlikte farklı cihazların, farklı renk modellerinin, yol bulma ve yol planlama algoritmalarının kullanımı kullanıcılara sağlayacaktır.

KAYNAKÇA

Anumba, C. J., Ugwu, O. O. ve Ren, Z. (Ed.) (2005). *Agents and multi-agent systems in construction*. A.B.D.: Taylor & Francis Pub.

Bourg, D. M. ve Seeman G. (2004). *AI for game developers*. A.B.D.: O'Reilly Media, Inc.

Brachman, R. J. ve Dietterich, T. (Ed.) (2007). *A concise introduction to multiagent systems and distributed artificial intelligence*. A.B.D.: Morgan & Claypool Pub.

Canming (b.t.). *EmguCV*, 15.03.2009, http://www.emgu.com/wiki/index.php/Main_Page

ColoRotate (b.t.). *Color Models*, 23.06.2009, http://learn.colorotate.org/color-models.html

Dichiera, I. (1999). *Robot soccer with global vision.* (tez)

Dijkstra, E. W. (1959). *A note on two problems in connexion with graphs*. Numerische Mathematik 1, 269 – 271

Gupta, G. S., Messom, C. H. ve Demidenko, S. (2005). Real-time identification and predictive control of fast mobile robots using global vision sensing. *IEEE Transactions on Instrumentation and Measurement*, 54 (1).

Hart, P. E., Nilsson, N.J. ve Raphael, B. (1968). A formal basis for the heuristic determination of minimum cost paths. *IEEE Transactions of Systems Science and Cybernetics*, Vol. SSC-4, No. 2, 100 - 107.

Keskin, O. (2008). *Çok ajanlı sistemlerde yol bulma algoritmaları ve uygulamaları*. (seminer notları)

Kordic, V., Lazinica, A. ve Merdan, M. (Ed.) (2005). *Cutting Edge Robotics* içinde (212 – 408). Hırvatistan: pro literatur Verlag.

Kirillov, A. (b.t.). *AForge.NET Features*, 15.03.2009, http://www.aforgenet.com/framework/features

Lego Mindstorms NXT, (b.t.). 16.05.2009, http://en.wikipedia.org /wiki/Lego_mindstorms_nxt

Lego (04.01.2006). What's NXT?, 16.05.2009, http://www.lego.com /eng/info/default.asp?page=pressdetail&contentid=17278

Liu, J. ve Wu, J. (2001). *Multi Agent Robotics*. A.B.D.: CRC Press

Twenga (b.t.). *LOGITECH Quickcam Pro for Notebooks Webcam*, 26.04.2009, http://www.twenga.co.uk/specs-Quickcam-Pro-for-Notebooks-LOGITECH-Webcam-126053

Marchand-Maillet, S. ve Sharaiha, Y. M. (2000). *Binary Digital Image Processing: A Discrete Approach*. İngiltere: Academic Press

Morris, J. (1998). *Data Structures and Algorithms*, 12 Nisan 2008, http://www.cs.auckland.ac.nz/software/AlgAnim/dij-op.html

NGI Book (2007). *Blob Analysis*, 12.03.2009, http://www.ngi-central.org/ngi_documentation/html/chunk/ch04s07.html

Perš, J. ve Kovacic, S. (2000). Computer Vision System for Tracking Players in Sports Games. *International Workshop on Image and Signal Processing And Analysis 2000*. Hırvatistan

Sallhammar, K. (2004). *Bluetooth Version 1.1 / IEEE 802.15.1* (ders notları)

Sidran, E. (2005). *An Analysis of Dimdal's "An Optimal Pathfinder for Vehicles in Real-World Terrain Maps".* (PhD tez)

Siciliano, B. ve Khatib, O. (Ed.) (2008). *Springer Handbook Of Robotics*. Almanya: Springer.

Skiena, S. S. (2008) *The Algorithm Design Manual* (2). İngiltere: Springer.

Smed, J. ve Hakonen, H. (2006). *Algorithms and Networking for Computer Games*. İngiltere: John Wiley & Sons Ltd.

Yüksel T. ve Sezgin A. (2006). Mobil Robotlar İçin Izgara Tabanlı Haritalarda En Kısa Yol Algoritmalarını Kullanarak Yol Planlama. *Otomatik Kontrol Türk Milli Komitesi Otomatik Kontrol Ulusal Toplantısı 2006*, 367 - 372

Wachi, K. (10 Ocak 2009). Re: FOV - EFL of 9000 Pro [55044] http://forums.logitech.com/logitech/board/print?board.id=quickcam_software&message.id=55044