PAMUKKALE ÜNİVERSİTESİ FEN BİLİMLERİ ENSTİTÜSÜ

ENGELLİ BİR ALAN İÇİNDE HEDEFİNE OTOMATİK OLARAK ULAŞABİLEN BİR MOBİL ROBOTUN KABİLİYETİNİN ARAŞTIRILMASI

YÜKSEK LİSANS TEZİ

Okan MİNNETOĞLU

Anabilim Dalı : Makine Mühendisliği

Programı : Makine Teorisi ve Dinamiği

Tez Danışmanı: Prof. Dr. Erdinç Şahin ÇONKUR

TEMMUZ 2013

YÜKSEK LİSANS TEZ ONAY FORMU

Pamukkale Üniversitesi Fen Bilimleri Enstitüsü 101111002 nolu öğrencisi Okan MİNNETOĞLU tarafından hazırlanan "**ENGELLİ BİR ALAN İÇİNDE HEDEFİNE OTOMATİK OLARAK ULAŞABİLEN BİR MOBİL ROBOTUN KABİLİYETİNİN ARAŞTIRILMASI**" başlıklı tez tarafımızdan okunmuş, kapsamı ve niteliği açısından bir Yüksek Lisans tezi olarak kabul edilmiştir.

Tez Danışmanı : (Jüri Başkanı)	**Prof. Dr. Erdinç Şahin ÇONKUR (PAÜ)**
Jüri Üyesi :	**Doç. Dr. Murat SARI (PAÜ)**
Jüri Üyesi :	**Yrd.Doç. Dr. Veysel ALKAN (PAÜ)**

Pamukkale Üniversitesi Fen Bilimleri Enstitüsü Yönetim Kurulu'nun 31/07/2013 tarih ve 25/16... sayılı kararıyla onaylanmıştır.

Fen Bilimleri Enstitüsü Müdürü
Prof. Dr. Nuri KOLSUZ

Bu tezin tasarımı, hazırlanması, yürütülmesi, araştırmalarının yapılması ve bulgularının analizlerinde bilimsel etiğe ve akademik kurallara özenle riayet edildiğini; bu çalışmanın doğrudan birincil ürünü olmayan bulguların, verilerin ve materyallerin bilimsel etiğe uygun olarak kaynak gösterildiğini ve alıntı yapılan çalışmalara atfedildiğine beyan ederim.

İmza : 

Öğrenci Adı Soyadı : Okan MİNNETOĞLU

ÖNSÖZ

Bu çalışmada engelli bir alan içinde hedefine otomatik olarak ulaşabilen bir mobil robotun tasarımı iyileştirilerek kabiliyeti arttırılmıştır. Daha sonra hareketsiz ve/veya hareketli engellerin bulunduğu ortamda mobil robotun engellere çarpmadan, engellerden kaçınarak hedefine ulaşabilmesini sağlamak amacıyla hareket algoritmaları geliştirilmiştir. Hareket algoritmaları geliştirilirken potansiyel alan metodundan yararlanılmıştır. Sadece sabit engellerin olduğu ortamlarda potansiyel alan metodu mobil robotun yol planlamasının yapılabilmesi için yeterli olmaktadır. Fakat hareketli bir tane bile engelin bulunduğu ortamda potansiyel alan metodunun kullanılmasının mobil robot yol planlaması için yeterli olmadığı gözlenmiştir. Bu çalışmada ortamdaki hareketli engelin veya engellerin konumlarının tespiti dikkate alınarak yol planlamasının yapıldığı algoritmalar geliştirilmiştir. Bu algoritmaların Microsoft Visual Studio NET kullanılarak uygulandığı yazılımlar geliştirilerek algoritmaların verimli olarak çalıştıkları gözlemlenmiştir. Ayrıca mobil robotun ani hız değişimlerine maruz kalmamasını sağlamak amacıyla mobil robotun ivmeli hareketi üzerine de çalışılmıştır. Bu çalışmanın gerçeklenmesinde katkıda bulunan yüksek lisans tez danışmanım Prof. Dr. E. Şahin ÇONKUR hocama teşekkür ederim. Teze maddi destek veren Pamukkale Üniversitesi Bilimsel Araştırma Projeleri Koordinasyon Birimine (PAUBAP Proje No: 2011FBE079) teşekkürlerimi sunarım. Son olarak da bana hem maddi hem de manevi destekte bulunan aileme teşekkürlerimi sunmak istiyorum.

Temmuz 2013

Okan Minnetoğlu
Makine Mühendisi

İÇİNDEKİLER

Sayfa

KISALTMALAR

ADC **:** Analog To Digital Converter (Analog Dijital Dönüştürücü)
ARM **:** Advanced RISC Machines (Gelişmiş RISC Cihazları)
CPU **:** Central Processing Unit (Merkezi İşlem Birimi)
FPU **:** Floating Point Unit (Kayan Nokta Ünitesi)
GPIO : General Purpose Input Output (Genel Amaçlı Giriş Çıkış)
ICC **:** Instantaneous Center of Curvature (Anlık Eğrilik Merkezi)
LCD **:** Liquid Crystal Display (Sıvı Kristal Gösterge)
LED **:** Light-Emitting Diode (Işık Yayan Diyot)
MB **:** Megabayt
MHz **:** Megahertz
PC **:** Personal Computer (Kişisel Bilgisayar)
PIC **:** Peripheral Interface Controller (Çevresel Arabirim Denetleyicisi)
RAM **:** Random Access Memory (Rastgele Erişimli Hafıza)
SD **:** Secure Digital (Güvenli Dijital)
UPS **:** Uninterruptable Power Supply (Kesintisiz Güç Kaynağı)
USB **:** Universal Serial Bus (Evrensel Seri Veriyolu)
V **:** Volt

TABLO LİSTESİ

Tablolar

ŞEKİL LİSTESİ

Şekiller

SEMBOL LİSTESİ

θ	Açısal hız
ϕ_r	Sağ teker hızı
ϕ_l	Sol teker hızı
u	Tekerlek yarıçapı
V_f	Doğrusal hız
l	Aks uzunluğu
R_{ICC}	Mobil robotun etrafında döndüğü nokta ile aks merkezi arasındaki mesafe
t_1	Mobil robotun toplam hareket süresi
tAcc	Mobil robotun pozitif ivmelenme zamanının toplam hareket zamanına olan oranı
tDec	Mobil robotun negatif ivmelenme zamanının toplam hareket zamanına olan oranı
rstep	Mobil roborun hızı

ÖZET

ENGELLİ BİR ALAN İÇİNDE HEDEFİNE OTOMATİK OLARAK ULAŞABİLEN BİR MOBİL ROBOTUN KABİLİYETİNİN ARAŞTIRILMASI

Mobil robot teknolojisi devamlı gelişmektedir. Günümüzde de mobil robotlar genellikle askeri, bilimsel ve endüstriyel araştırma yapması amacıyla kullanılmaktadır. Bu çalışmada engelli bir alan içinde hedefine otomatik olarak ulaşabilen bir mobil robotun tasarımı iyileştirilerek kabiliyeti arttırılmıştır ve hareket algoritmaları geliştirilmiştir. Mobil robotun toplam ağırlığını ve maliyetini azaltmak için masaüstü bilgisayar yerine Mini 2440 isimli ARM işlemcisine sahip olan gömülü bir sistem kullanılmıştır. Güç kaynağı olarak UPS yerine de binek araçlarda kullanılan 12V'luk 72 amperlik bir akü tercih edilmiştir. Mobil robotta hareket kontrol kartı yerine STM32F4 Discovery Board kullanılarak mobil robotun hareket kabiliyeti arttırılmıştır.

Mobil robotun hareketin başlangıç ve hedef konumu belli bir ortamda hareketli ve/veya hareketsiz engellerden kaçınarak hedefe ulaşılabilmesi için Visual C# .NET programında bir yazılım geliştirilmiştir. Bu yol planlamasında potansiyel alan metodundan faydalanılmıştır. Potansiyel alan çalışma sahasında hareketli engeller olduğu durumda yetersiz kalmaktadır. Bu hareketli engellerin konumlarının dikkate alınarak mobil robot yol planlamasının yapılabilmesi için yeni bir algoritma geliştirilmiştir. Ayrıca rastgele ekranda beliren engellerin dikkate alınması ve çarpma riski taşımayan engellerin dikkate alınmaması gibi özellikler algoritmaya eklenmiştir. Daha sonra bu algoritma sadeleştirilerek Mini 2440 gömülü sisteme adapte edilmiştir. Mini 2440 gömülü sistem ile STM32F4 Discovery Board arasında RS-232 haberleşmesi yapılarak mobil robotun planlanan yolda hareketi sağlanmış ve engellerden kaçınarak hedefine ulaştığı gözlemlenmiştir.

Anahtar Kelimeler: Mobil robot, yol planlama, potansiyel alan metodu, engellerden kaçınma, sabit engeller, hareketli engeller

SUMMARY

RESEARCHING THE ABILITY OF A MOBILE ROBOT REACHING AUTOMATICALLY TO THE GOAL IN A CLUTTERED ENVIRONMENT

Mobile robot techonology is constantly evolving. Today, the mobile robots are often used to for military purposes, for scientific and industrial researches. In this work the ability of a mobile robot reaching automatically to the goal in a cluttered environment by improving its design has been increased and motion algorithms have been implemented. To reduce the total weight and cost of the mobile robot Mini 2440 embedded system with an ARM processor is used instead of a desktop computer. As a power supply 12V, 72 ampers battery used in passengers cars has been preferred to UPS. The motion abilility of the mobile robot has been increased using STM32F4 Discovery Board instead of a motion control card.

A software program has been developed using Visual C# .NET programming language to be able to reach its goal by avoiding obstacles in an cluttered environment with moving and/or static obstacles. Potential field method is utilised for the path planning. The potential field method is not sufficient in the case of moving obstacles in the workspace. A new algorithm has been developed to be able to achieve the path planinig of the mobile robot regarding the positions of these moving obstacles. Besides, some features have been added to the algorithm such as taking into account random obstacles and ignoring the obstacles that do not pose a threat. Later on, this algorithm has been adopted to the embedded system by means of its simplification. The movement of mobile robot on the planned path has been made by communicating Mini 2440 embedded system and STM32F4 discovery board using RS-232 and it has been observed that the mobile robot reaches its destination by avoiding obstacles.

Key Words: Mobile robot, path planning, potential field method, avoiding obstacles, stationary obstacles, moving obstacles

1. GİRİŞ

Mobil robotların kullanım alanları gün geçtikçe artmaktadır. Bu kullanımın artmasında mobil robot teknolojisinin gelişmesi önemli bir rol oynamaktadır. Mobil robotlar günümüzde genellikle askeri, bilimsel ve endüstriyel araştırma yaparak insanlara yardımcı olmaktadır. Mayın tespiti ve insansız hava araçları askeri araştırma yapan, deniz altı ve uzay araştırmaları yapan robotlar da bilimsel araştırma yapan mobil robotlara örnek verilebilir.

1.1 Tezin Amacı

Engelli bir alanda hedefine varabilen mobil robotun tasarımının iyileştirilerek kabiliyetinin arttırılması, mobil robotun hareketsiz ve/veya hareketli engellerin bulunduğu ortamda engellere çarpmadan hedefine ulaşabilmesi için takip etmesi gereken yolun planlandığı bir algoritmanın geliştirilmesi ve tasarlanan mobil robotun bu algoritma kullanılarak planlanan yolu takip ederek engellere çarpmadan hedefine ulaşmasını sağlamaktır.

1.2 Literatür Özeti

Literatürde mobil robot konusunda çok fazla çalışma yapılmıştır. Çalışmalar mobil robotla ilgili çok çeşitli alanlara yayılmıştır. Çalışmaların bir kısmı yol planlaması üzerinedir. Diğer kısım çalışmalara ise mobil robotta davranış temelli bulanık kontrol uygulanarak ve görüntü işleme kullanılarak mobil robotun hedefine ulaşmasını sağlayan çalışmalar örnek verilebilir.

Mobil robotların otomatik bir şekilde hedefine varabilmeleri için yol planlama algoritmalarının geliştirilmesi gerekmektedir. Yol planlama algoritmaları mobil robotların engeller arasından engellere çarpmadan hedeflerini otomatik olarak bulmasını hedeflemektedir. Yol planlama algoritmalarını 3 ana gruba ayırabiliriz:

1)Yol haritalaması: Alanda bulunan engelsiz kısımlarda bir takım yollar oluşturulur.

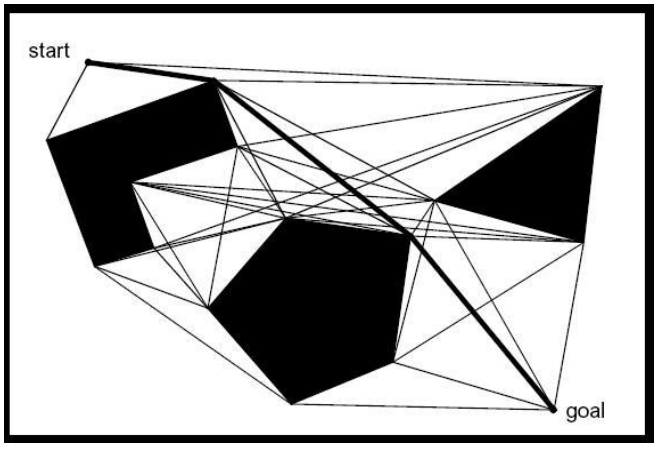
a)Görünürlük Grafiği

b)Voronoi Diyagramı

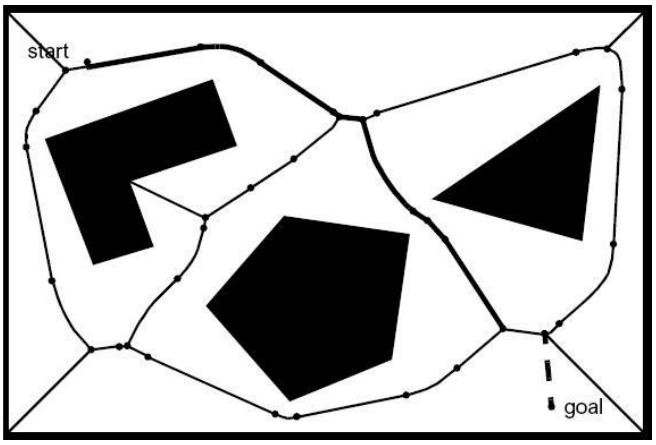
2)Hücre ayrıştırması: Alan parçalara bölünerek yol planlaması gerçekleştirilir

3)Potansiyel alan: Matematiksel bir fonksiyon, alan üzerine uygulanır.

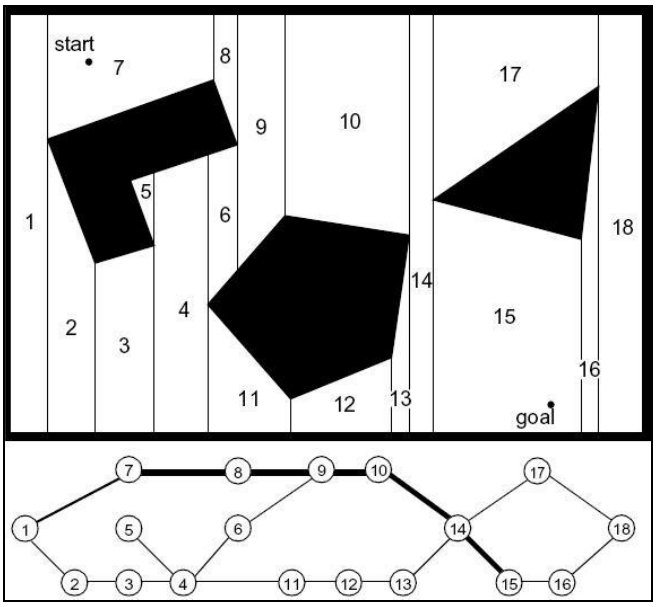
Engelsiz bir ortamda başlangıç noktası ile hedef noktası arasına çizilen düz yol mobil robotun yolunun planlanması için yeterli olmaktadır. Fakat engellerin olduğu bir ortamda yol planlaması bu kadar kolay olmamaktadır. Görünürlük grafiğinde [1] engellerin köşe noktaları da dikkate alınarak başlangıç ve hedef noktası arasında düz yollar çizilerek yol planlaması yapılmaktadır. Planlanan yol üzerindeki hareket sırasında engellere çarpışmaması için çizilen yolların üstünde engelin bulunmaması gerekmektedir (Şekil 1.1). Çizilen düz yollardan en kısa olanı ise en uygun yol olmaktadır. Voronoi diyagramında [1] ise yollar, engellerden eşit uzaklıkta olacak şekilde çizilmektedir ve yine aynı şekilde bu yolların en kısa olanı en uygun yol olmaktadır (Şekil 1.2). Hücre ayrıştırması yönteminde [1] Şekil 1.3'deki gibi alan hücrelere ayrıştırılarak yol planlaması yapılmaktadır. Hücreler ayrıştırıldıktan sonra komşu hücreler üzerinden giderek mobil robot hedefine ulaşmaktadır. Bu hareket boyunca mobil robotun, hücrenin hangi tarafından geçeceği önemli değildir.



Şekil 1.1 : Görünürlük grafiği

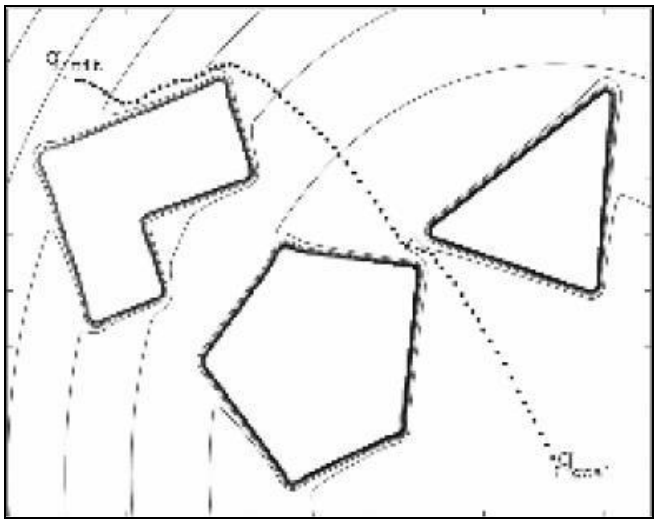


Şekil 1.2 : Voronoi diyagramı

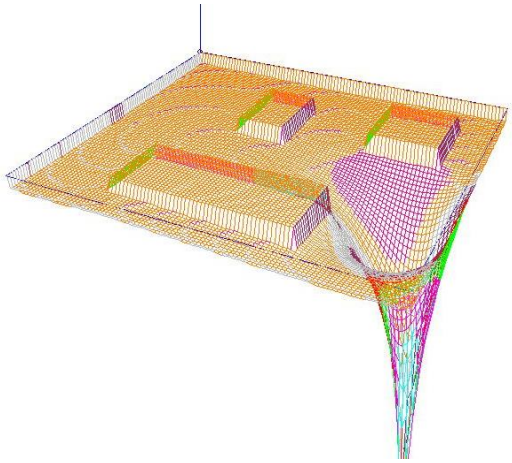


Şekil 1.3 : Hücre ayrıştırması

Potansiyel alan metodunda [2] engellerin mobil robot üzerinde itme, hedefin ise çekme etkisi yarattığından bahsedilmiştir. Bu iki etkinin sonucunda mobil robot engellerden kaçınarak hedefine ulaşmaktadır (Şekil 1.4). Bu metodun uygulanmasında potansiyel alan denklemi tüm çalışma alanına uygulanmaktadır. Bu uygulama sonucunda [3] hedef çukurda kalmaktadır, engeller tepe etkisi yaratmaktadır ve alanda hedefe doğru bir eğim oluşmaktadır (Şekil 1.5).

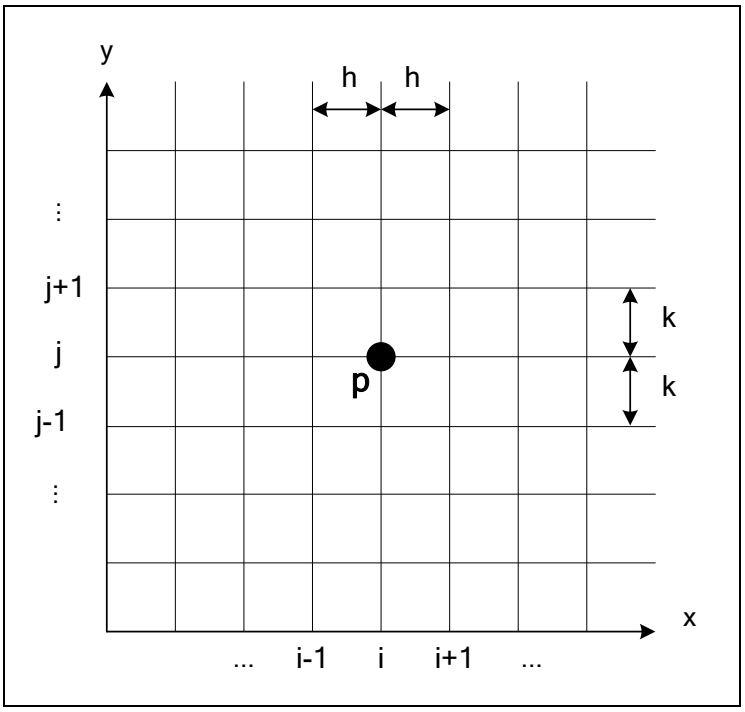


Şekil 1.4 : Potansiyel alanda mobil robotun hedefini bulması



Şekil 1.5 : Potansiyel alanın üç boyutlu görüntüsü

Potansiyel alan metodunun matematiksel ifadesi sonlu farklar yönteminden yararlanılarak elde edilebilir. İkinci mertebeden kısmi diferansiyel denklemler sonlu farklar metodu ile çözülmek istendiğinde, çözüm uzayı bir ızgara sistemiyle (grid system) ifade edilir. Genellikle ızgara sistemi olarak dikdörtgensel bir ızgara sistemi kullanılmaktadır (Şekil 1.6). P noktasındaki ikinci mertebeden merkezi farkları x ve y'ye göre (1.1) ve (1.2)'deki eşitlikler ile ifade edilebilir.



Şekil 1.6 : P noktasının iki boyutlu bir alanda gösterilmesi

$$\frac{\partial u_{i,j}^2}{\partial x^2} = \frac{u_{i+1,j} - 2u_{i,j} + u_{i-1,j}}{h^2} \tag{1.1}$$

$$\frac{\partial u_{i,j}^2}{\partial y^2} = \frac{u_{i,j+1} - 2u_{i,j} + u_{i,j-1}}{k^2} \tag{1.2}$$

İki boyutlu Laplace denklemi aşağıdaki verilmiştir.

$$\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = 0 \tag{1.3}$$

(1.1) ve (1.2) denklemlerini (1.3) denkleminde yerine koyarsak;

$$\frac{u_{i+1,j} - 2u_{i,j} + u_{i-1,j}}{h^2} + \frac{u_{i,j+1} - 2u_{i,j} + u_{i,j-1}}{k^2} = 0 \tag{1.4}$$

(1.4) denklemi elde edilir. Eğer h = k olursa denklem aşağıdaki hali alır:

$$u_{i+1,j} + u_{i-1,j} + u_{i,j+1} + u_{i,j-1} - 4u_{i,j} = 0 \tag{1.5}$$

veya

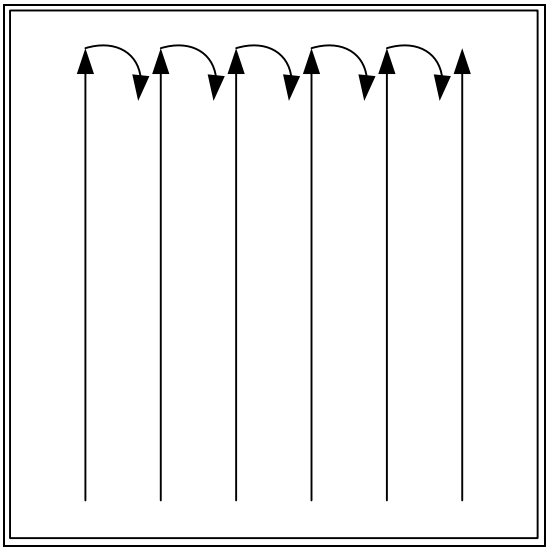
$$u_{i,j} = \frac{u_{i+1,j} + u_{i-1,j} + u_{i,j+1} + u_{i,j-1}}{4} \tag{1.6}$$

(1.6) denklemi potansiyel alan denklemidir. Eğer alandaki sınır değerleri biliniyorsa bu denklem Dirichlet sınır koşulları altındaki potansiyel alan denklemidir.

Çalışma alanını çevreleyen grid noktalarına ve engelleri temsil eden grid noktalarına sıfır değeri, hedef noktasına ise -2^{124} değeri verilmektedir. En dıştaki grid noktaları, engellere ait grid noktaları ve hedefe ait grid noktası dışındaki grid noktaları potansiyel alan denklemi ile hesaplanmaktadır. Bir iterasyon yapıldığında bu bahsedilen grid noktaları bir defa hesaplanmaktadır. İterasyon sayısı kadar grid noktaları hesaplanmaktadır.

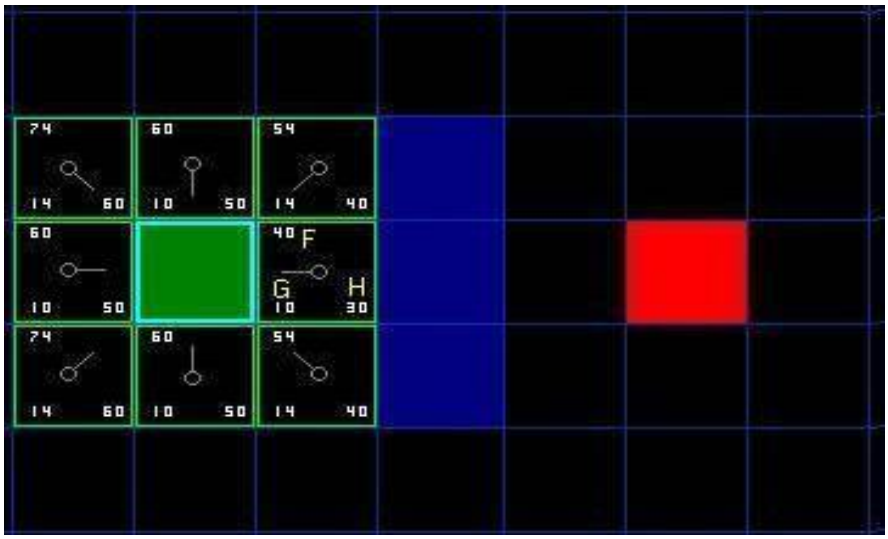
$$u_{i,j}^{m+1} = \frac{1}{4}(u_{i-1,j}^{m} + u_{i+1,j}^{m} + u_{i,j-1}^{m} + u_{i,j+1}^{m}) \tag{1.7}$$

(1.7) denklemindeki m iterasyon sayısını göstermektedir. Şekil 1.7'de alanın hesaplanma şekli gösterilmektedir. Şekil 1.7'de aşağıdan yukarıya doğru hesaplama gösterilmiştir. Hesaplama yönü yukarıdan aşağıya, sağdan sola veya soldan sağa şeklinde de seçilebilir. Eğer alanın indis değerleri (0, 0)'dan başlıyorsa (i, j) indisi (1, 1)'den başlamaktadır. Alanın en büyük indis değerleri (n, n) ise (n-1, n-1) indisli grid noktasına kadar hesaplama yapılmaktadır.



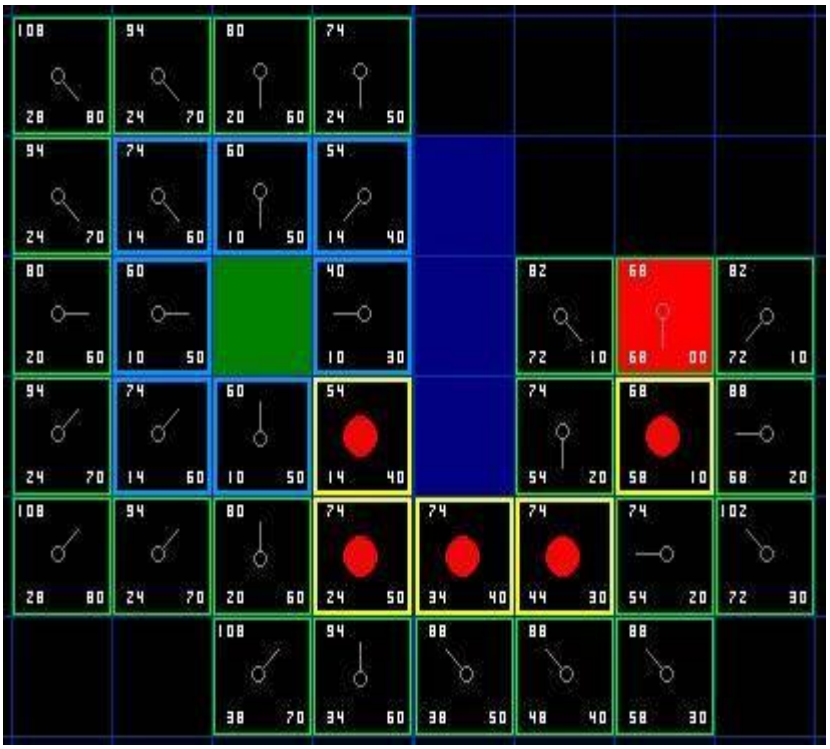
Şekil 1.7 : P noktasının iki boyutlu bir alanda gösterilmesi

Yol planlaması yapılırken en kısa yolun tespit edilebilmesi için A* algoritması [4] geliştirilmiştir. Bu algoritmada hareket alanı düğümlere ayrılarak her bir düğüme ulaşılabilmesi için katedilmesi gereken G mesafeleri ve o düğümden hedefe varılabilmesi için katedilmesi gereken H mesafeleri hesaplanmaktadır. Hesaplanan bu iki mesafe değeri toplanarak F mesafe değeri elde edilmektedir.



Şekil 1.8 : Yol planlaması için gerekli olan mesafelerin hesabı [5]

Şekil 1.8'de görüldüğü gibi başlangıç yeşil, engel mavi ve hedefte kırmızı renk ile gösterilmiştir. Başlangıç noktasından komşu düğümlere ulaşılabilmesi için katedilmesi gereken mesafeler G, düğümden hedefe ulaşılabilmesi için katedilmesi gereken mesafe değerleri de H ile isimlendirilmektedir. Düğümler arası geçişte düz bir adım 10 birim ve çarpraz adımda 14 birimdir. Hedefe ulaşırken alınması gereken yol bir düğüm için hesaplanırken ortadaki engel dikkate alınmamaktadır. Örneğin başlangıç noktasının sağındaki düğüm bir adım yakınında olduğu için G değeri 10'dur. Bu düğümden hedefe en kısa yoldan ulaşabilmek için ise sağa doğru üç adım gitmek gerekmektedir. Engel yokmuş gibi farzedildiğinde H değeri de 30 olmaktadır. Bu iki değer toplanarak F değeri 40 olmaktadır. Bu hesaplar her bir düğüm için yapıldıktan sonra F değerinin en küçük olduğu düğümden harekete devam edilerek ortada bulunan engelden de kaçınarak hedefe ulaşması için takip edilmesi gereken en kısa yolun tespiti Şekil 1.9'daki gibi yapılmıştır. Düğümlerden hedefe ulaşılabilmesi için birden fazla yol olacağı için bu algoritma sezgisel bir algoritma olarak isimlendirilmektedir ve engellerin hareket etmediği ortamlara uygulanabilmektedir.



Şekil 1.9 : En kısa yolun tespiti [5]

Anthony Stentz [6-7] tarafından engellerin hareketli olduğu ortamlarda da kullanılabilecek olan D* algoritması geliştirilmiştir ve A* algoritmasına [4] benzediği için D* olarak adlandırılmıştır. D* algoritmasında da A* algoritmasına benzer işlemler yapılmaktadır. Farklı olarak hareketli engellerin olduğu dinamik bir ortamda düğümlerden hedefe ulaşılabilmesi için katedilmesi gereken mesafe değerleri devamlı olarak güncellenerek engellerden kaçınılarak hedefe varılması sağlanmaktadır. Bu algoritma kullanıldığında devamlı olarak değerler güncellendiği için yol planlama işleminin tamamlanması uzun sürmektedir.

Bu eksikliği gidermek amacıyla yine Anthony Stentz [8] tarafından Focused D* algoritması geliştirilmiştir. D* algoritması kullanılarak yapılan yol planlaması işleminin uzun sürme eksikliği bu algoritma ile giderilmeye çalışılmıştır. İşlem süresi öncelikli olarak ortamın karmaşıklığına bağlıdır. Diğer bir deyişle ortamdaki engellerin sayısına, boyutuna, yerleşim şekline bağlıdır. Dolayısıyla işlem süresindeki azalma ortalama bir değerdir.

Sinir ağı yönteminde [9] biyolojiden esinlenilerek gridler nöronlara, tüm alan da topolojik düzende olan sinir ağına benzetilmiştir . Her nöronun sadece lokal bağı ve bir de manevra denklemi (shunting equation) bulunmaktadır. Bu denklem robotun bir sonraki konumunu belirlemekte kullanılmaktadır. Statik ortamda değerler zamanla

değişmemekte fakat dinamik ortamda zamanla değişime uğramaktadır. Hesabın karmaşıklığı sinir ağının boyutu ile doğrusal olarak artmaktadır.

Path transform yönteminde [10] transform fonksiyonu kullanılarak engellere çarpmadan robotun hedefe varması sağlanmaktadır. Bu yöntemde iki kontrol noktası seçilerek bu iki kontrol noktası için işlemler yapılmaktadır ve ilaveten engellere çarpılıp çarpılmayacağı tespit edilerek yol planlaması yapılmaktadır. Bu yöntem sabit engellerin bulunduğu ortamlarda uygulanabilmektedir ve sezgisel bir yöntemdir.

Sinir ağı [9] ve path transform [10] yöntemlerinden esinlenilerek robotun hedefine ulaşmasına sağlayabilecek bir algoritma [11] daha geliştirilmiştir. Bu algoritmada sinir ağı yönteminden farklı olarak hesaplamalar tüm çalışma alanına yayılmıştır. Path transform yönteminden farklı olarak bu algoritma hareketli engellerin, hedeflerin olduğu dinamik ortamda da robotun hedefine varmasını sağlayabilmektedir.

Daha sonra geliştirilen yol planlama algoritmasına [11] eklemeler yapılarak engel ile çarpışmaların daha etkili tespit edilebildiği bir başka algoritma [12] geliştirilmiştir. Bu algoritma D* algoritmasına benzemektedir fakat bu algoritmada hareketli engellerin olduğu dinamik bir ortamda düğümlerden hedefe ulaşılabilmesi için katedilmesi gereken mesafe değerleri devamlı olarak güncellenmemektedir. Engellerin etrafında güvenli sınırlar oluşturularak ve ceza fonksiyonları kullanılarak bu algoritmada hareketli engellerin olduğu dinamik ortamlarda da robotun hedefine ulaşması sağlanmaktadır.

2. HAREKET ALGORİTMALARININ GELİŞTİRİLMESİ

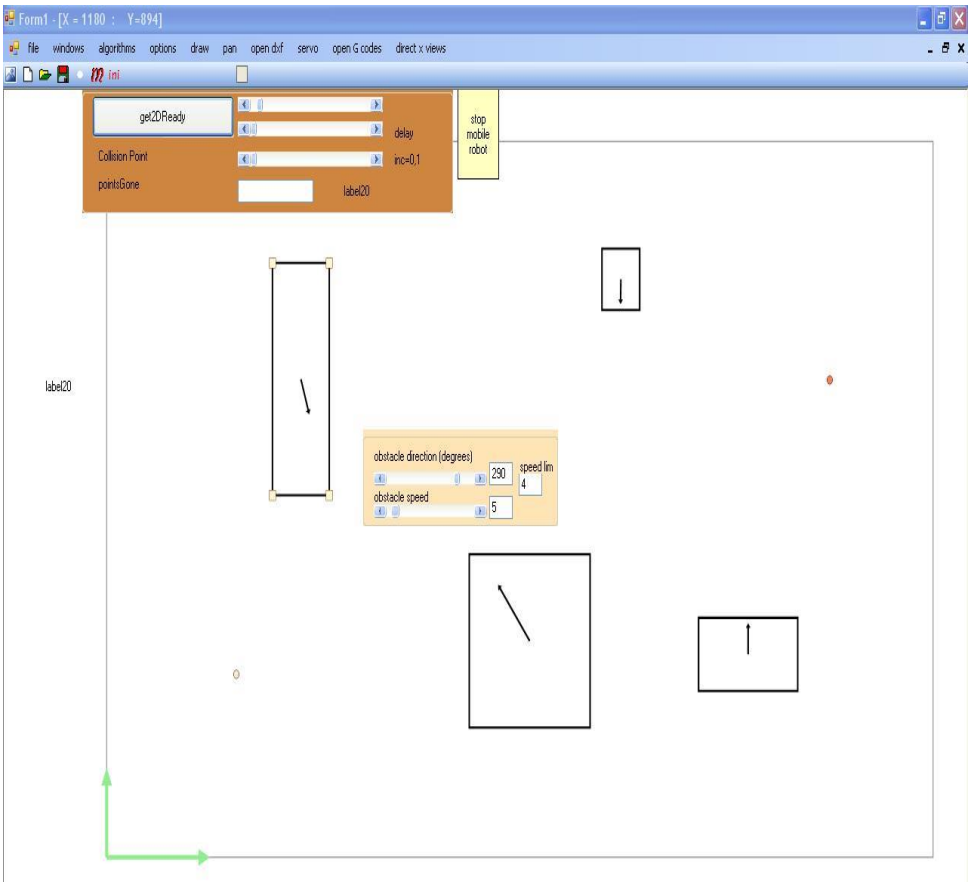
Algoritma geliştirilmesi bu tezin ana çalışmalarından biri olduğu için bu konuda detaylı çalışmalar yapılmıştır. Engelsiz bir ortamda mobil robot, başlangıç noktasından hedef noktasına doğru çizilen düz bir çizgi boyunca hareketine devam ederek hedefine ulaşabilmektedir. Engellerin sabit, yani olduğu yerde hareketsiz bir biçimde bulunduğu bir ortamda potansiyel alan metodu ile yapılan hesaplamalar yardımı ile mobil robot, engellere çarpmadan hedefine ulaşabilmektedir. Fakat ortamda hareketli engeller olduğu zaman bu hareketli engellerin konumlarının dikkate alınarak mobil robotun yol planlamasının yapılması gerekmektedir. Engeller devamlı hareket halinde oldukları için mobil robotun hareketi sırasında çarpışmalar gerçekleşebilmektedir. Bu durumu engelleyebilmek için engellerin konumlarının tespiti ve bu tespitin kullanılarak mobil robotun hareketli engellere de çarpmadan hedefine ulaşabilmesini sağlayan bir algoritma geliştirilmiştir.

2.1 Ana Algoritmanın Geliştirilmesi

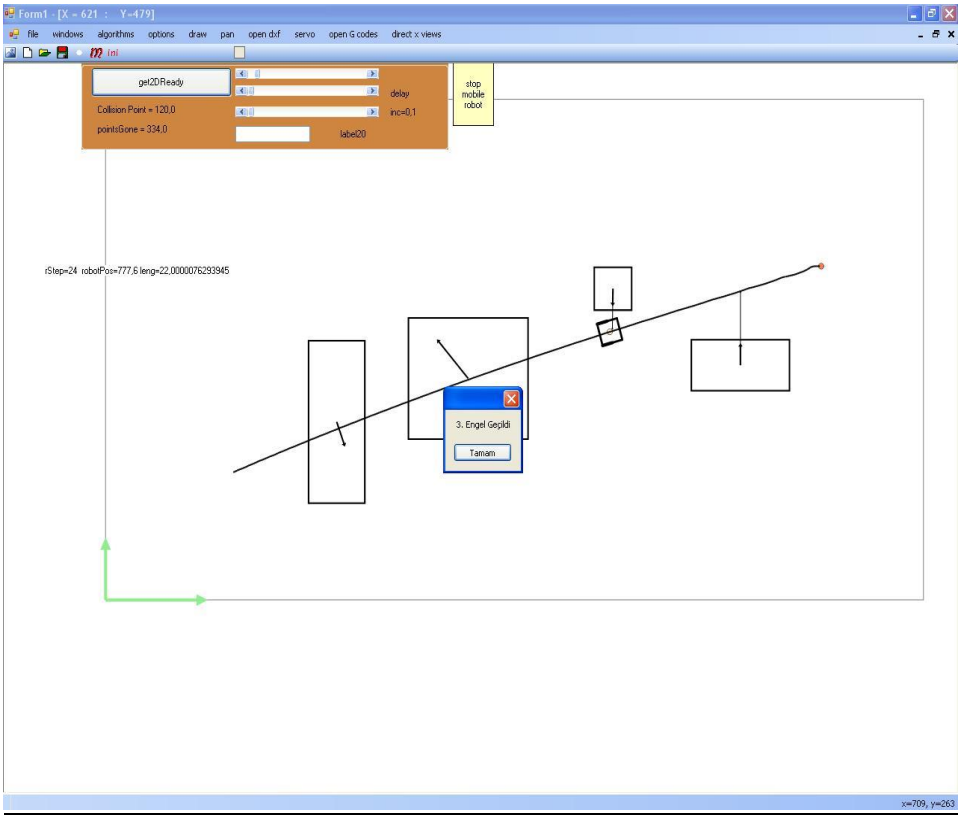
Öncelikli olarak potansiyel alan metodundan yararlanılarak ana algoritma geliştirilmiştir. Geliştirilen bu ana algoritmanın üzerine eklemeler yapılarak devamlı olarak algoritmaya yeni özellikler kazandırılmıştır. Sadece sabit engellerin olduğu ortamlarda potansiyel alan metodu mobil robotun yol planlamasının yapılabilmesi için yeterli olmaktadır. Fakat hareketli bir tane bile engelin bulunduğu ortamda potansiyel alan metodu kullanıldığı zaman optimum yol bulunamamaktadır. Çünkü engelin her hareketinde potansiyel alan yeniden hesaplanmaktadır ve bu durum yolun devamlı olarak hedeften uzaklaşmasına hatta bazen de ileri-geri hareket yapmasına sebep olmaktadır. Dolayısıyla algoritmada ortamdaki hareketli engelin veya engellerin konumları dikkate alınmıştır ve bu algoritma kullanılarak yol planlamasının yapıldığı bir yazılım Microsoft Visual Studio NET kullanılarak geliştirilmiştir. Yazılımın kısaca açıklanması gerekirse bu yazılımda ekrana fare yardımıyla engeller çizilmekte ve bu engellere hareket doğrultuları ve hızları verilebilmektedir. Hareketli engellere çarpmadan mobil robotun hedefine ulaşması sağlanmaktadır. Çizilen engeller nokta olarak düşünülmüş, dikdörtgen engelin

yükseklik ve genişlik değeri ihmal edilmiştir. Mobil robotun hedefine doğru, yolunu bulmasında potansiyel alan metodu kullanılmıştır. Potansiyel alan metodunun matematiksel ifadesinin elde edilmesi detaylı bir şekilde [13] de açıklanmıştır.

Özetle ana algoritmadan bahsedilmesi gerekirse engelin hızı dikkate alınarak mobil robotun engeli geçebilmesi için gerekli hızı hesaplanarak engeller geçilmektedir. Algoritmada 1. engel geçilene kadar 2. engel ile ilgili bir hesaplama yapılmamaktadır. Algortimada 1. engelin geçildiği anlaşıldıktan sonra mobil robotun 2. engeli geçebilmesi için ihtiyaç duyduğu hız hesaplanarak bu şekilde mobil robotun engelleri geçerek hedefine ulaşması sağlanmaktadır. Bu algoritma kullanılarak hesaplanan hızlar bazen çok yüksek çıkmakta ve mobil robotun ulaşamayacağı hızlara çıkmasını gerektirmektedir. Bu yüzden bundan sonraki algoritmada hızlara sınır verilerek bu sorun ortadan kaldırılmaya çalışılmıştır. Şekil 2.1'de ekrana engeller çizilip bu engellere hareket doğrultuları ve hız değerleri verilmiştir. Şekil 2.2'de ise mobil robotun, hedefine ulaşması için izlemesi istenen yol belirlenmiş ve kaçıncı engelin geçildiği mesaj kutusu ile kullanıcıya bildirilmiştir.



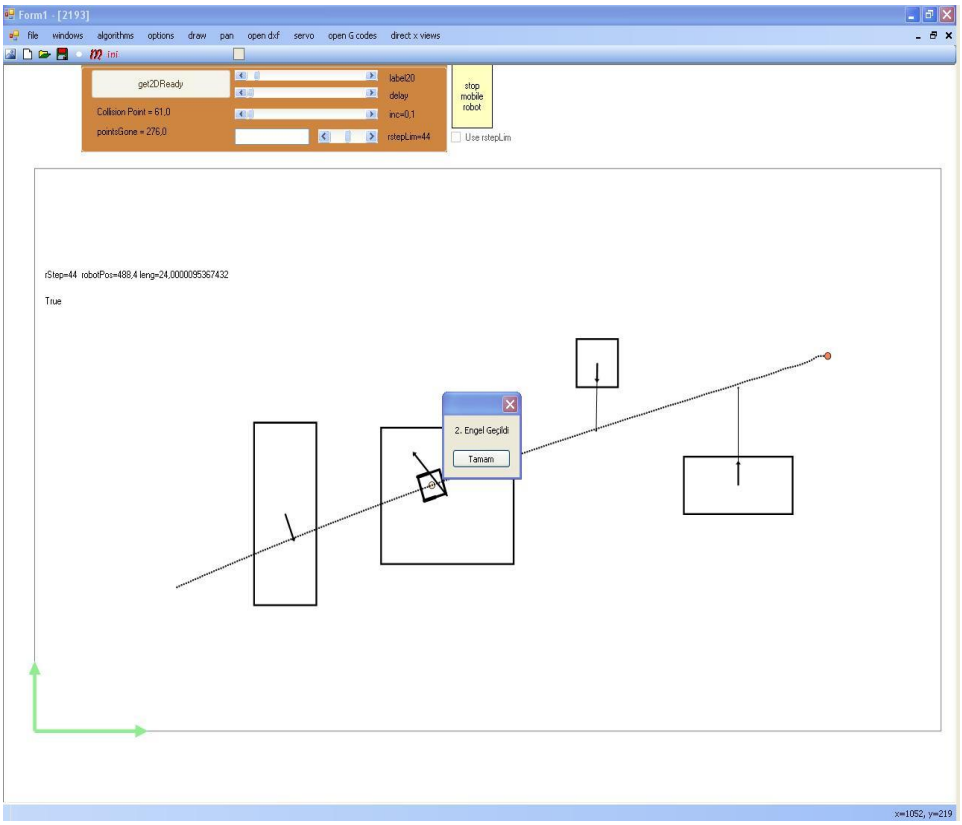
Şekil 2.1 : Yazılımda engellerin hareket doğrultusunun ve hızının belirlenmesi



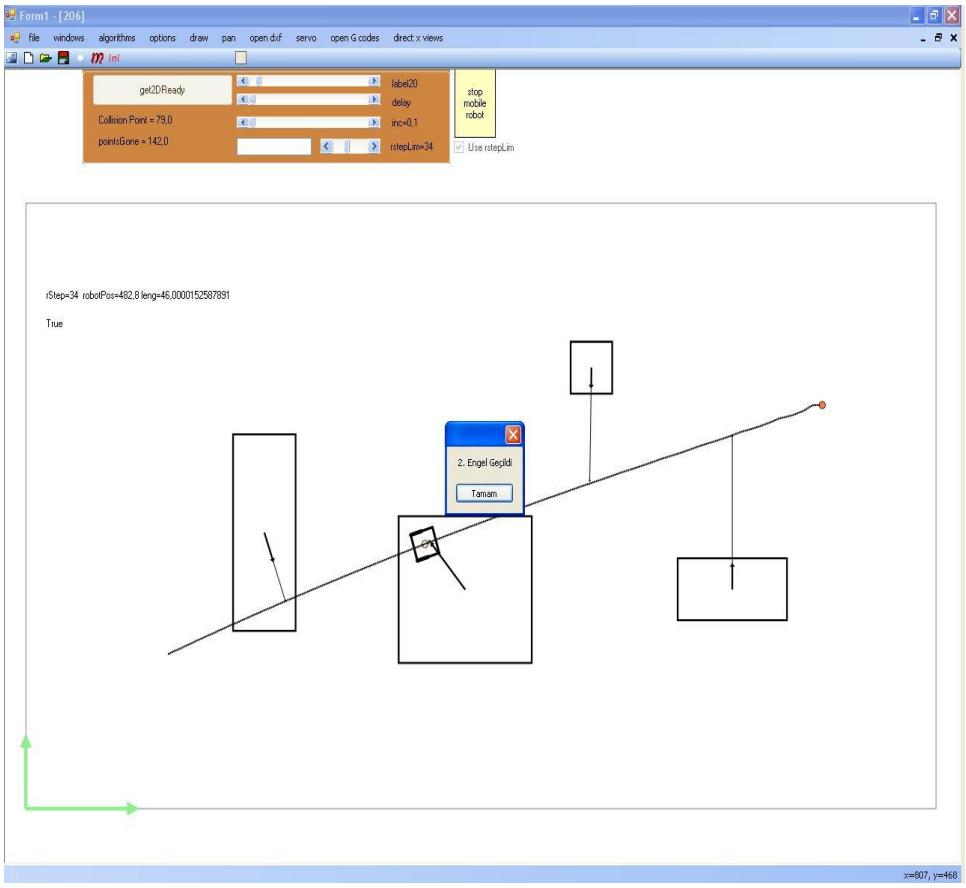
Şekil 2.2 : Engelin geçildiğinin mesaj kutusu ile kullanıcıya bildirilmesi

2.2 Yazılımdaki Maksimum Hız Değerinin Sınırlandırılması

Bu yazılımda mobil robotun ulaşamayacağı kadar yüksek hızlarla hareket etme problemi hıza sınır verilerek çözülmeye çalışılmıştır. Mobil robotun hızı rstep ve hız sınırı da rstepLim isimli değişkenlerle ifade edilmiştir. Mobil robotun hızına hscrollbar yardımı ile sınır değeri verilince mobil robotun engele çarpma riski oluşmuştur. Bu yüzden engele çarpıp çarpmayacağının tespiti üzerine çalışılmıştır. Bu durumun tespiti engele çok yakın mesafede iken mobil robotun ve engelin hızlarının dikkate alındığı yeni bir hesap yapılarak sağlanmıştır. Engele çarpılacaksa mobil robotun durup engelin geçmesini beklemesi algoritması üzerine çalışmaların temelleri atılmıştır. Ayrıca Use rstepLim ismindeki checkbox seçildiğinde hız sınır değerinde iken sabit hızla engellere çarpmadan hedefine ulaşması sağlanmıştır. Şekil 2.3'de görüldüğü gibi hız sınırı hscrollbar yardımıyla 44 olarak ayarlanarak mobil robotun ikinci engeli geçerken hızının 44 piksel olması sağlanmıştır. Şekil 2.4'de ise hız sınırı 34 olarak ayarlandıktan sonra Use rstepLim ismindeki checkbox seçilmiştir ve mobil robotun 2. engeli geçerken hızının 34 piksel olduğu gözlenmiştir.



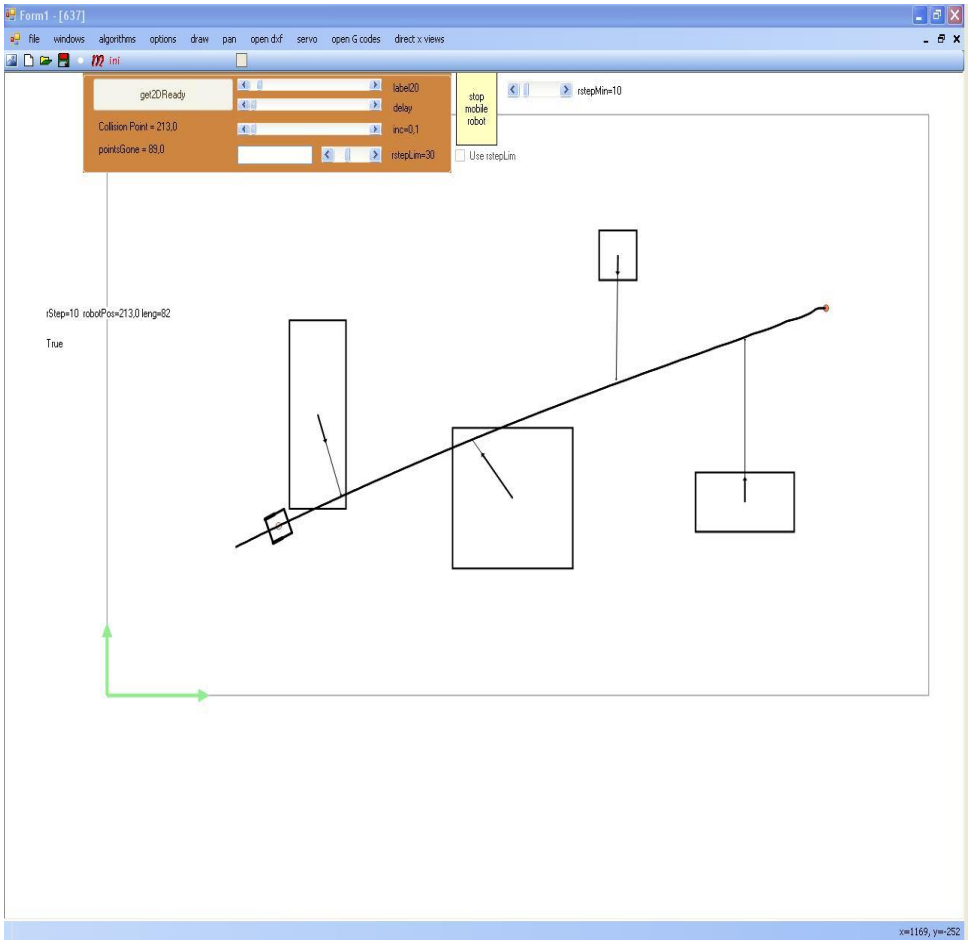
Şekil 2.3 : Mobil robotun hız sınırının hscrollbar yardımıyla ayarlanması



Şekil 2.4 : Checkbox yardımıyla mobil robotun sabit hızlı hareketinin sağlanması

2.3 Kod Optimizasyonu

Mobil robotun mümkün olduğunca hızlı bir şekilde hedefine ulaşabilmesi için yazılıma, minimum hıza alt sınır verilmesini sağlayan bir özellik eklenmiştir. Hızın alt sınır değeri rstepMin isimli hscrollbar ile ayarlanabilmektedir. Hızın çok düşük çıktığı durumlarda mobil robotun alt sınır değerindeki hızla hareket etmesi sağlanmıştır. Hızın sınırlanmasından dolayı mobil robotun engellere çarpıp çarpmayacağının anlaşılması ve engele çarpılacaksa mobil robotun durup engelin geçmesini beklemesinin sağlanması için detay çalışmaları yapılmıştır. Ayrıca kod optimizasyonu yapılarak daha sonraki geliştirmeler yapılırken kod karmaşasından kurtulmak amaçlanmıştır. Mobil robotun hızındaki ani değişikliklerin pratik uygulamalarda çok kullanışlı olmayacağı düşünülerek ivmeli hareket üzerinde çalışma yapılmıştır. Şekil 2.5'de görüldüğü gibi hızın alt sınır değeri hscrollbar yardımıyla 10 olarak ayarlandıktan sonra mobil robotun birinci engele doğru ilerlerken hızının 10 piksel olduğu görülmektedir.

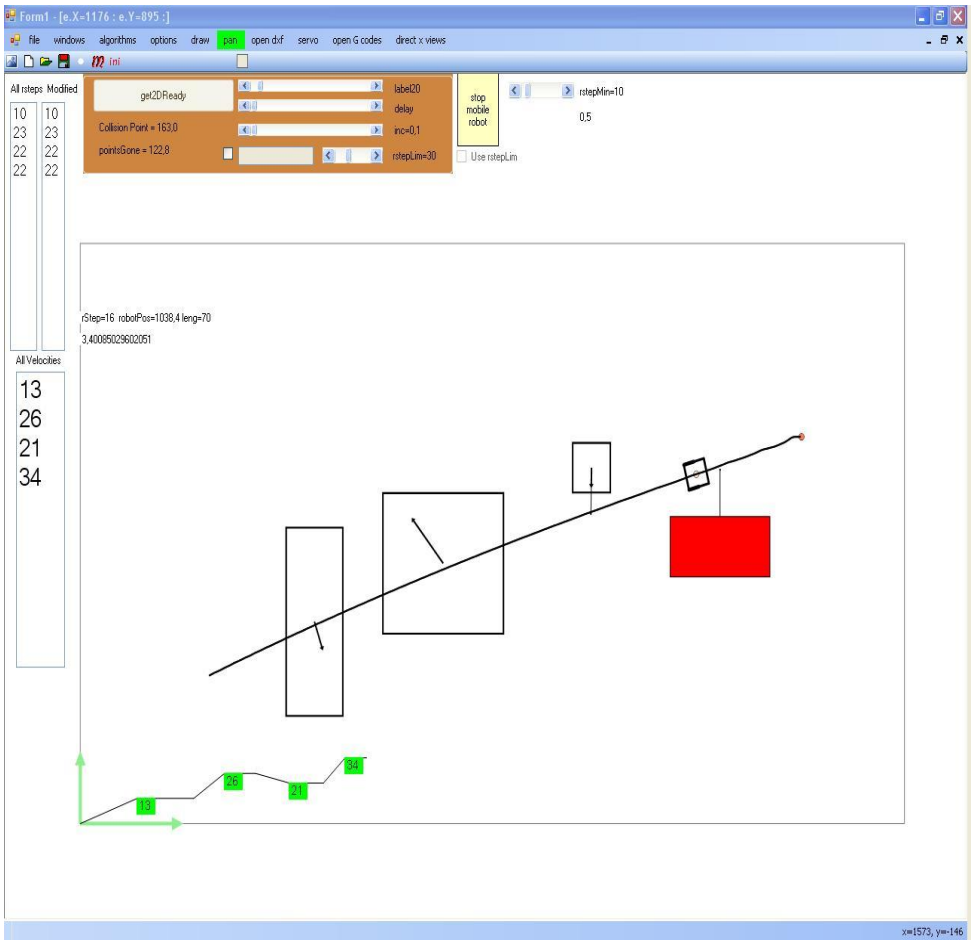


Şekil 2.5 : Mobil robotun hızın alt sınır değeri ile hareketi

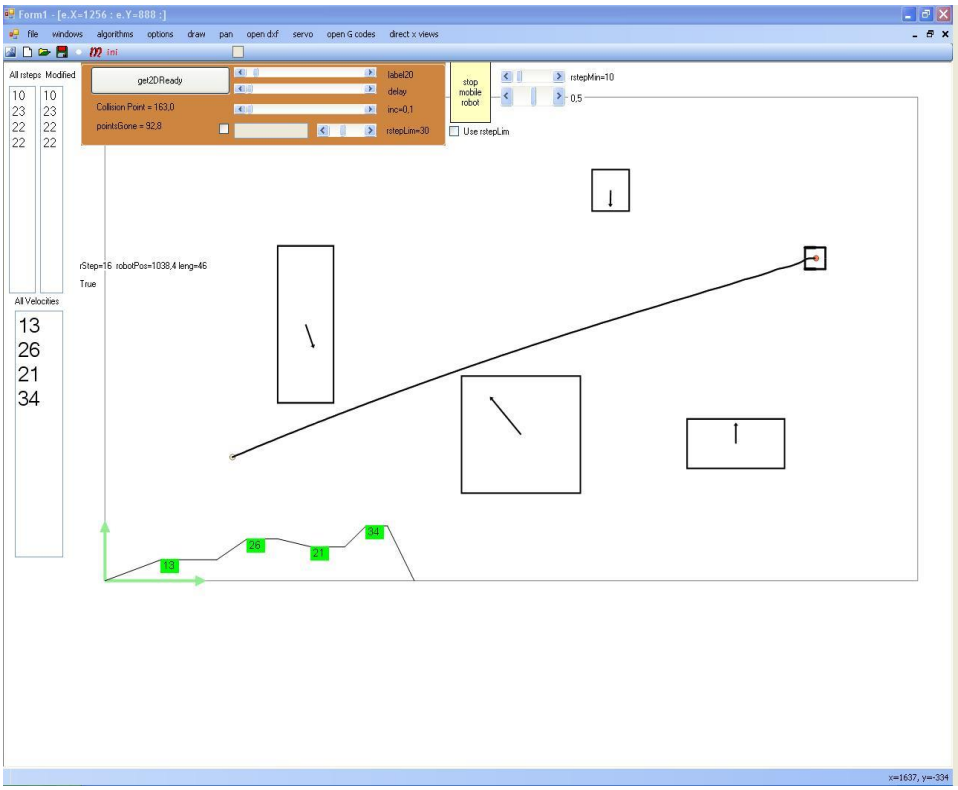
2.4 Yazılıma İvmeli Hareketin Eklenmesi

Mobil robotun ani hız değişimlerine maruz kalmaması için, yazılıma gerekli kod eklemeleri yapılarak mobil robotun ivmelenerek hareket etmesi sağlanmıştır. Mobil robotun sabit hız ile hareket ederken katetmesi gereken mesafe ile ivmelenerek hareket ederken katetmesi gereken mesefenin aynı olması gerektiği gerçeğine dayandırılarak hesaplamalar ve bu hesaplamalar kullanılarak kod eklemeleri yapılmıştır. Bir de yazılımın işlevselliğinin arttırılması için çalışmalar yapılmaya başlanmıştır. Örneğin engel çizimlerinin kolay yapılabilmesi için klavyedeki r tuşu kullanılmıştır. Fare ile ekran yaklaştırıldıktan (zoom) sonra a tuşu ile ekrandan uzaklaştırılması (zoom out) sağlanmış ve n tuşu ile yeni dosya oluşturulması sağlanmıştır. Şekil 2.6'da görüldüğü gibi mobil robotun hareketinde de geçilmeye çalışılan engel kırmızı renge boyanarak yazılımın görselliğinin arttırılması amaçlanmıştır.

İvmeli hareketin algoritması sabit hız ile hareket ederken gidilen mesafenin doğrusal olarak artan bir hız ile gidilmek istendiğinde hızın her adımda ne kadar değiştiği temeline dayanmaktadır. Şekil 2.7'de görüldüğü gibi bir tane hscrollbar yardımı ile ivmelenmenin hareketin hangi anına kadar gerçekleşeceği ayarlanabilmektedir ve ona göre hızlar hesaplanmaktadır. Bu hesaplar dikkate alınarak hız-zaman grafiği çizim ekranının sol alt köşesinde çizilmiştir. Hareket ile grafiğin birbiriyle uyumlu olması amacıyla grafiğin üzerine kırmızı bir daire çizilmiştir. Bu daire hareket boyunca grafik üzerinde hareket ederek kullanıcıya hareket hakkında bilgi vermektedir. Fakat yazılımda yapılan uygulamalarda çizilen grafik ile mobil robotun hareketinin birbiri ile tam uyumlu olmadığı, kullanıcıya verilen bilgilerin yanlış olduğu tespit edilmiştir. Bu uyumsuzluğun engel geçildikten sonra yeniden hesaplandıktan sonra ortaya çıkan hız değişiminden dolayı robotun bir adımda alınan mesafelerin eşit olmamasından kaynaklandığı belirlenmiştir. Daha sonra detaylıca anlatılacağı üzere bu uyumsuzluk sorunu çözülmüştür.



Şekil 2.6 : Geçilmeye çalışılan engelin kırmızıya boyanması



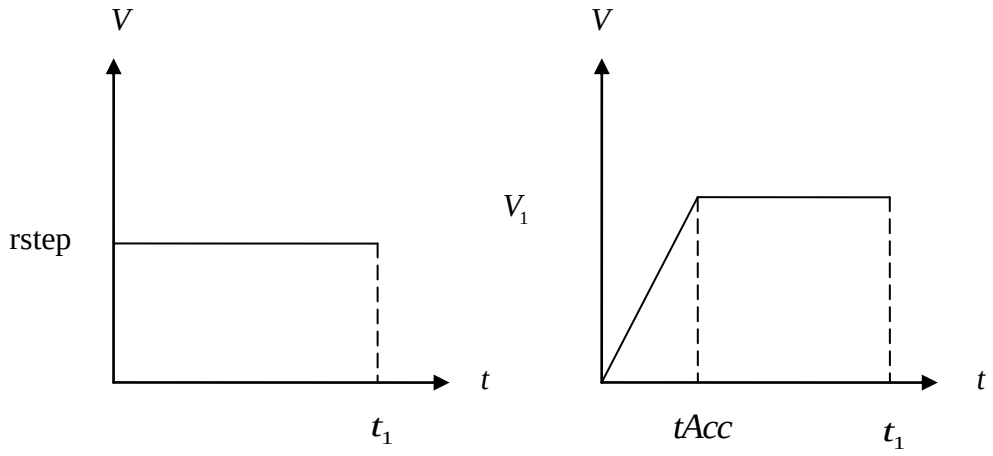
Şekil 2.7 : Mobil robotun ivmeli hareketinin grafiği

2.5 Doğrusal Hareketten İvmeli Harekete Geçişte Hız Hesaplamaları

Bu hesaplamalar, daha önce de belirtildiği gibi doğrusal ve ivmeli hareket ile mobil robotun gideceği toplam mesafelerin eşit olması gerektiği gerçeğine dayandırılarak yapılmıştır. Hız-zaman grafiğinin altındaki alanların toplamı gidilen toplam mesafe olduğuna göre bu eşitlik yardımıyla ivmeli harekette mobil robotun hız-zaman grafiği kolaylıkla çizilmiştir. Mobil robotun hareketi 3 kısma ayrılarak hesaplamalar yapılmıştır.

2.5.1 Başlangıç hareketi

Başlangıç hareketi, mobil robotun harekete başladığı kısımdır. Şekil 2.8'de de görüldüğü üzere rstep mobil robotun hızı, t_1 mobil robotun hareket süresi, $tAcc$ ise mobil robotun ivmelenmesinin, toplam hareketin tamamlandığı zamanın %'de kaçında gerçekleştiğidir. Örneğin $tAcc = 0{,}5$ ise ve mobil robotun hedefine ulaşması 10 saniye sürüyorsa, mobil robotun 5. saniye de ivmelenip daha sonra da 5 saniye boyunca sabit hızla hareketine devam etmesi gerekmektedir. V_1 ise doğrusal hareketten ivmeli harekete geçişteki, mobil robotun ivmelenirken ulaşması gereken maksimum hız değeridir.



Şekil 2.8 : a) Doğrusal hareket hız-zaman grafiği b) İvmeli hareket hız-zaman grafiği

Mobil robotun başlangıç hareketinde grafiklerin altında kalan alanların birbirine eşit olması gerektiği gerçeğine dayandırılarak hesaplamalar yapılmıştır. Şekil 2.8b'de görüldüğü gibi başlangıç hareketinin grafiği kesik çizgilerle belirtilmiş olan bir tane üçgen ve bir tane dikdörtgenden oluşmaktadır. Grafiklerin altında kalan alanlar birbirlerine eşitlenerek (2.5) genel eşitliği elde edilir.

$$\frac{V_1.tAcc}{2} + V_1.(t_1 - tAcc) = rstep.t_1 \tag{2.1}$$

$$V_1.tAcc + 2.V_1.t_1 - 2.V_1.tAcc = rstep.t_1.2 \tag{2.2}$$

$$2.V_1.t_1 - V_1.tAcc = rstep.t_1.2 \tag{2.3}$$

$$V_1.(2.t_1 - tAcc) = rstep.t_1.2 \tag{2.4}$$

$$V_1 = \frac{rstep.t_1.2}{2.t_1 - tAcc} \tag{2.5}$$

Örnek olarak ivmelenmenin toplam hareketin %10'luk kısmı boyunca gerçekleştiği farz edilmiştir. Bu varsayım sonucunda $tAcc = 0.1t_1$ eşitliği kullanılmıştır. (2.5) denkleminde $tAcc$ ifadesinin eşiti kullanılarak (2.6) denklemi elde edilir.

$$V_1 = \frac{rstep.t_1.2}{2.t_1 - 0.1t_1} \tag{2.6}$$

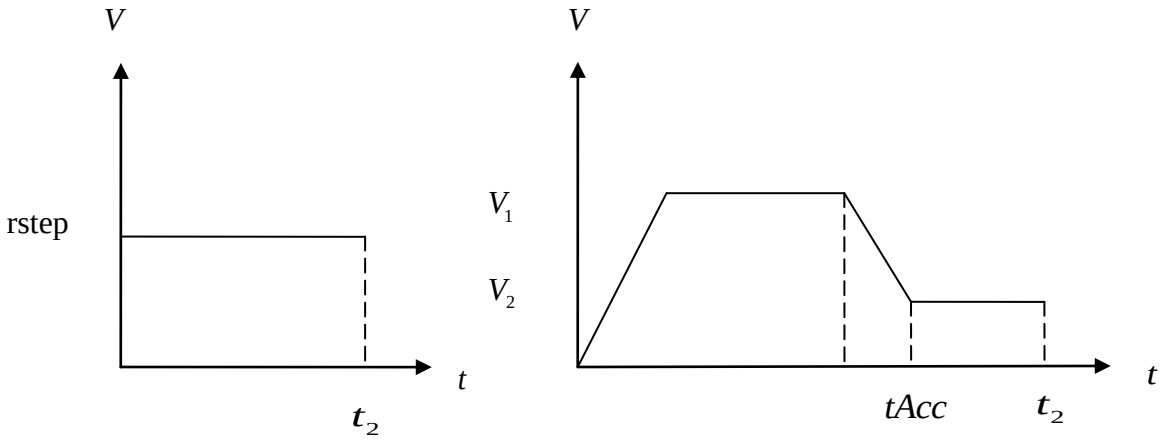
Daha genel olarak (2.7) denklemi elde edilir.

$$V_1 = \frac{rstep.2}{2 - tAcc} \tag{2.7}$$

2.5.2 Ara hareketi

Ara hareketi, mobil robotun başlangıç hareketi ile negatif ivmelendiği son hareketi arasında kalan kısımdır. Başlangıç hareketinde de bahsedildiği üzere rstep mobil robotun hızı, t_2 mobil robotun hareket süresi, $tAcc$ ise mobil robotun ivmelenmesinin, toplam hareketin tamamlandığı zamanın %'de kaçında gerçekleştiğidir. V_1 değeri başlangıç hareketindeki formül ile hesaplanmış olan hız değeridir. V_2 değeri ise mobil robotun ivmelenip V_1 hızına ulaşıp birinci engeli

geçtikten sonra ikinci engeli geçmesi için ulaşması gereken hız değeridir (Şekil 2.9b).



Şekil 2.9 : a) Doğrusal hareket hız-zaman grafiği b) İvmeli hareket hız-zaman grafiği

Mobil robotun ara hareketininde de grafiklerin altında kalan alanların birbirine eşit olması gerektiği gerçeğine dayandırılarak hesaplamalar yapılmıştır ve (2.15) denklemi elde edilmiştir. Şekil 2.9b'de görüldüğü gibi ara hareketin grafiği kesik çizgilerle belirtilmiş olan birer tane dikdörtgen ve yamuktan oluşmaktadır.

$$\left(\frac{V_1+V_2}{2}\right).tAcc+V_2.(t_2-tAcc)=rstep.t_2 \tag{2.8}$$

$$(V_1+V_2).tAcc+2.V_2(t_2-tAcc)=2.rstep.t_2 \tag{2.9}$$

$$V_1.tAcc+V_2.tAcc+2.V_2.t_2-2.V_2.tAcc=2.rstep.t_2 \tag{2.10}$$

$$V_1.tAcc-V_2.tAcc+2.V_2.t_2=2.rstep.t_2 \tag{2.11}$$

$$2.V_2.t_2-V_2.tAcc=2.rstep.t_2-V_1.tAcc \tag{2.12}$$

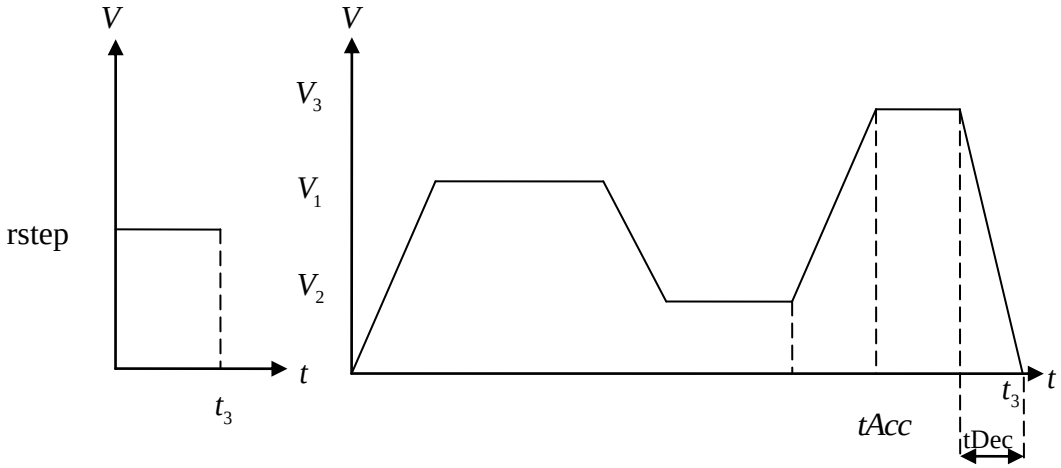
$$V_2.(2.t_2-tAcc)=2.rstep.t_2-V_1.tAcc \tag{2.13}$$

$$V_2=\frac{2.rstep.t_2-V_1.tAcc}{2.t_2-tAcc} \tag{2.14}$$

$$V_2=\frac{2.rstep-V_1.tAcc}{2-tAcc} \tag{2.15}$$

2.5.3 Son hareketi

Son hareketi, mobil robotun negatif ivmelenme yapmak suretiyle hedefine ulaştığı kısımdır. Başlangıç ve ara hareketinde de bahsedildiği üzere rstep mobil robotun hızı, t_3 mobil robotun hareket süresi, $tAcc$ ise mobil robotun ivmelenmesinin, toplam hareketin tamamlandığı zamanın %'de kaçında gerçekleştiğidir. V_1 değeri başlangıç hareketindeki formül ile hesaplanmış olan hız değeridir. V_2 değeri de yine benzer şekilde ara hareketindeki formül ile hesaplanmış olan hız değeridir. V_3 değeri mobil robotun negatif ivmelenmeden önce ulaşması gereken hız değeridir. tDec ise mobil robotun negatif ivmelenmesinin, toplam hareketin tamamlandığı zamanın %'de kaçında gerçekleştiğidir.



Şekil 2.10 : a)Doğrusal hareket hız-zaman grafiği b)İvmeli hareket hız-zaman grafiği

Mobil robotun hareketinin son kısmında da grafiklerin altında kalan alanların birbirine eşit olması gerektiği gerçeğine dayandırılarak hesaplamalar yapılmıştır. Şekil 2.10b'de görüldüğü gibi son hareketin grafiği kesik çizgilerle belirtilmiş olan birer tane yamuk, dikdörtgen ve üçgenden oluşmaktadır.

$$\left(\frac{V_2+V_3}{2}\right).tAcc+(t_3-tAcc-tDec).V_3+\frac{V_3.tDec}{2}=rstep.t_3 \tag{2.16}$$

$$(V_2+V_3).tAcc+2.V_3(t_3-tAcc-tDec)+V_3.tDec=2.rstep.t_3 \tag{2.17}$$

$$V_2.tAcc+V_3.tAcc+2.V_3.t_3-2.V_3.tAcc-2.V_3.tDec+V_3.tDec=2.rstep.t_3 \tag{2.18}$$

$$V_2.tAcc - V_3.tAcc - V_3.tDec + 2.V_3.t_3 = 2.rstep.t_3 \tag{2.19}$$

$$V_2.tAcc + V_3.(2.t_3 - tAcc - tDec) = 2.rstep.t_3 \tag{2.20}$$

$$V_3 = \frac{2.rstep.t_3 - V_2.tAcc}{2.t_3 - tAcc - tDec} \tag{2.21}$$

ivmelenme ile negatif ivmelenme zamanının birbirine eşit olduğu durum (2.22) incelenerek (2.24) denklemi elde edilir.

$$tAcc = tDec \tag{2.22}$$

$$V_3 = \frac{2.rstep.t_3 - V_2.tAcc}{2.t_3 - 2.tAcc} \tag{2.23}$$

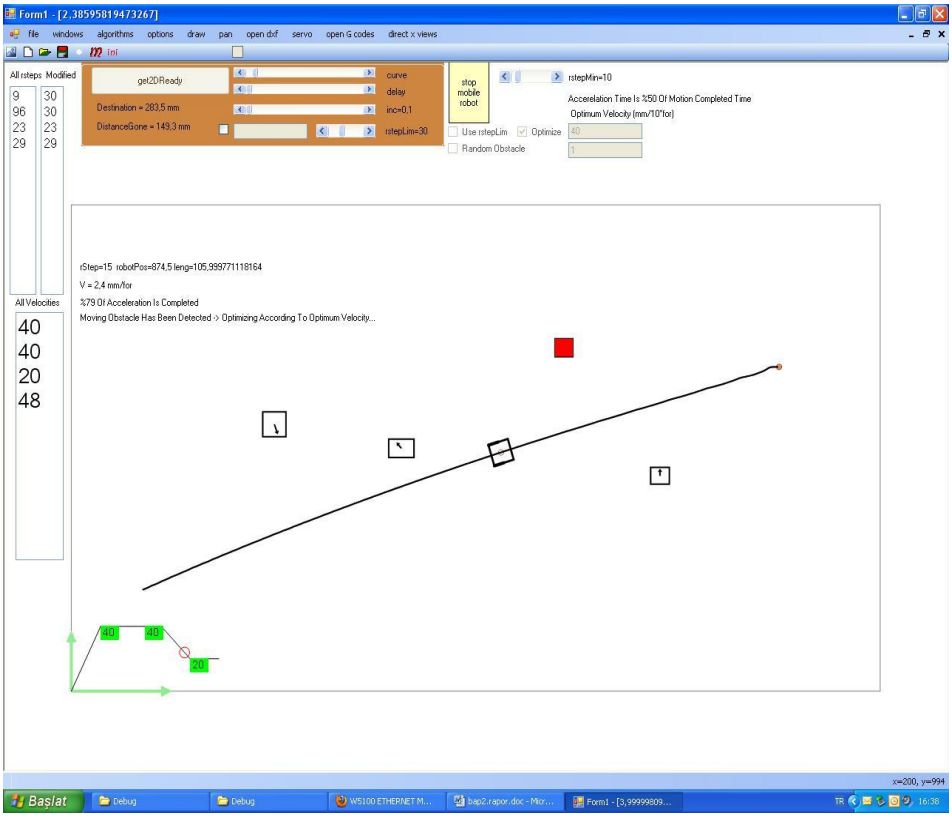
$$V_3 = \frac{2.rstep.t_3 - V_2.tAcc}{2.(t_3 - tAcc)} \tag{2.24}$$

Hareketin daha önceki kısımlarında detaylıca açıklandığı üzere 2.24 denklemi genelleştirilerek (2.25) denklemi elde edilir.

$$V_2 = \frac{2.rstep - V_2.tAcc}{2.tAcc} \tag{2.25}$$

2.6 Yazılımda Hareket-Grafik Senkronizasyonunun Sağlanması

Yazılımdaki çizilen hız grafiği ile mobil robotun hareketinin birbiri ile tam uyumlu olmaması sorunu ele alınmıştır. Bu sorunun çözülebilmesi için nasıl bir algoritma gerektiği üzerinde detaylı bir çalışma yapılmış daha sonra da bu algoritmanın kodları yazılarak yazılımdaki bu sorun çözülmüştür. Özetle algoritmada mobil robotun engeli ne kadar zamanda geçeceği tam olarak hesaplanıp aynı zamanda hareketin o anında ivmelenip ivmelenmediğinin bilinmesi gerekmektedir. Bu bilgiler yardımıyla hareketin ilgili yerinde Şekil 2.11'de görüldüğü gibi grafiğin üzerine kırmızı bir daire çizilerek hareket ile grafiğin senkronize olduğu kullanıcıya gösterilmiştir.

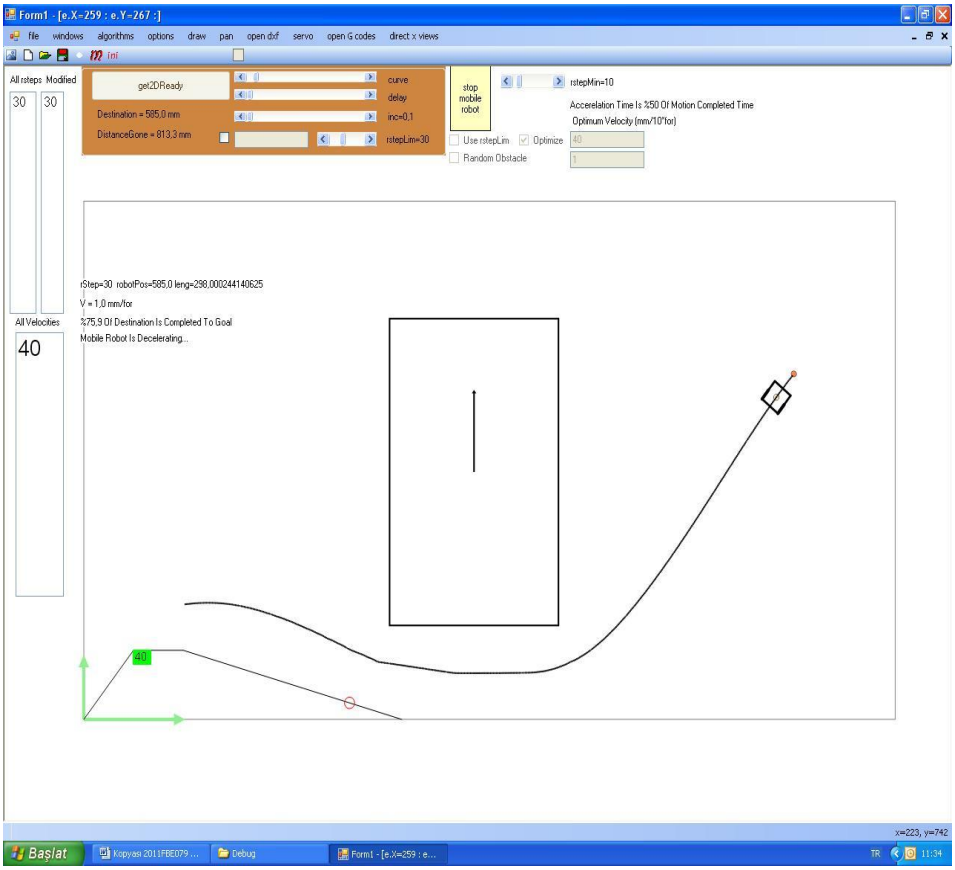


Şekil 2.11 : Mobil robotun hareketi ile grafik arasındaki senkronizasyon

Bu sorun çözüldükten sonra yazılımdaki mobil robotun ivmeli hareket grafiği incelendiğinde mobil robotun çok ani hız değişimlerine maruz kaldığı saptanmıştır. Bunun sebebi yazılımda engel engel hesaplama yapılmasıdır. Bu durum robotun ani hız değişimleri yapmasına sebep olmaktadır. Ani hız değişimlerini engellemek ve mobil robotun optimum bir hızda hedefine varmasını sağlamak amacıyla çalışmalar yapılmıştır. Çalışmaların sonucunda kullanıcıdan bir optimum hızın istendiği ve girilen bu optimum hıza göre minimum ve maksimum hızların belirlendiği ve bu hızlar aralığında engelleri geçebiliyorsa optimum hız ile mobil robotun engellerden kaçınarak hedefine varması sağlanmıştır. Buna ek olarak yazılımın dinamik ortamlarda da engelleri geçerek hedefini bulması amacıyla mobil robotun hareketi sırasında ekrana konumunun, yerinin, doğrultusunun ve hızının bilgisayarın seçtiği rastgele değerlerden oluşan engellerin eklendiği bir kod yapısı eklenmiştir. Dolayısıyla mobil robotun ivmeli hareketi gerçekleşirken rastgele bir engelle karşılaşmaktadır ve yapılan denemeler sonucunda mobil robotun engellerden kaçınarak hedefe vardığı gözlenmiştir.

2.6.1 Sabit bir engelin geçilmesi

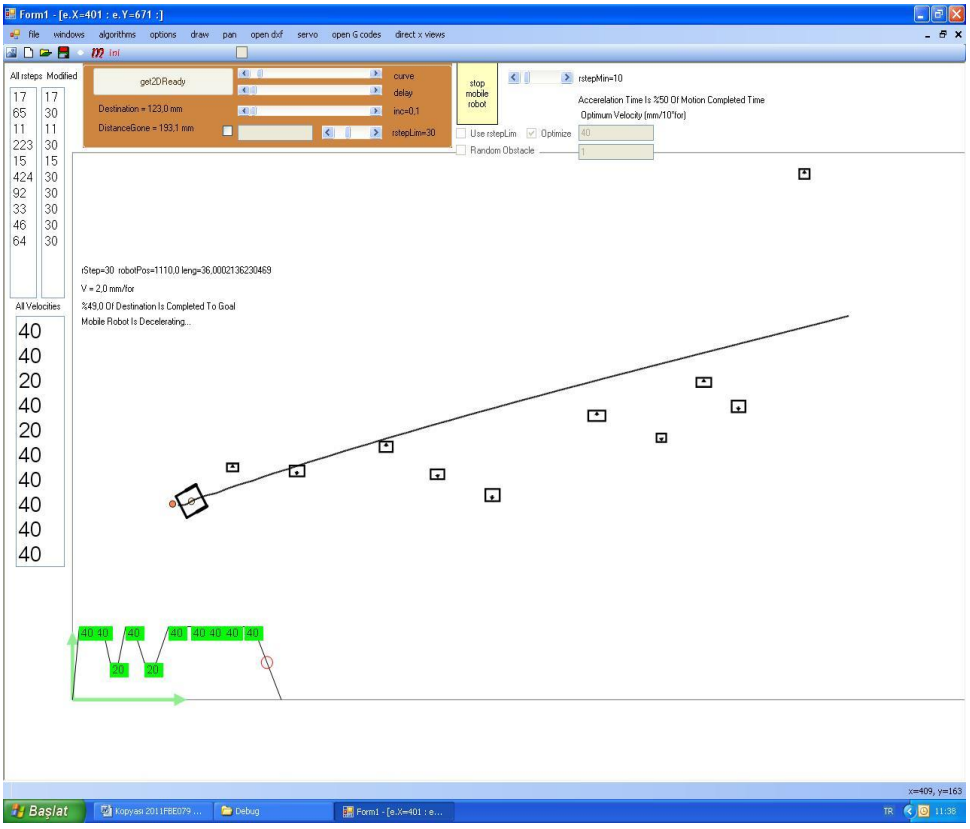
Yazılımda Şekil 2.12'de görüldüğü gibi ekrana sabit bir engel çizildikten sonra 'Optimize' isimli checkbox işaretlenerek istenilen hızda mobil robotun hareketi sağlanmış ve sabit olan engeli geçerek hedefe vardığı gözlenmiştir. Yazılımda geçilecek olan engelin sabit mi yoksa hareketli mi olduğu anlaşılıp daha sonra engel sabit ise engelin yükseklik ve genişliği dikkate alınarak mobil robotun izleyeceği yol belirlenmektedir.



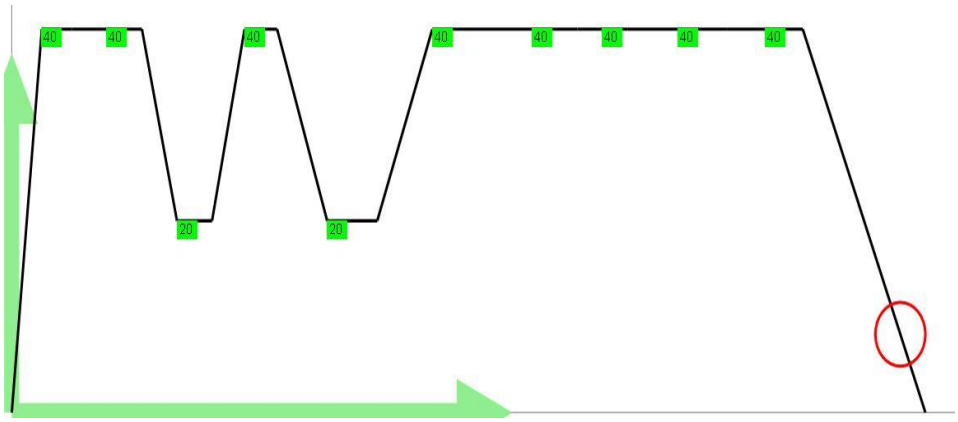
Şekil 2.12 : Sabit engelin geçilmesi

2.6.2 Hareketli 10 engelin geçilmesi

Yazılımdaki eksiklikleri görmek amacıyla değişik ortamlarda mobil robotun hareketleri ve engele varıp varmadığı incelenmiştir. Bu durumda robotun hareketine ekranın sağ tarafından başlaması ve ekranın sol tarafındaki hedefe varması istenmiştir. Ayrıca on tane engel çizilerek bu engellere farklı hızlar verilmiştir. Şekil 2.13'de görüldüğü gibi mobil robot engellere çarpmadan hedefine ulaşmıştır. Mobil robotun hareketinin hız-zaman grafiği de Şekil 2.14'de verilmiştir.



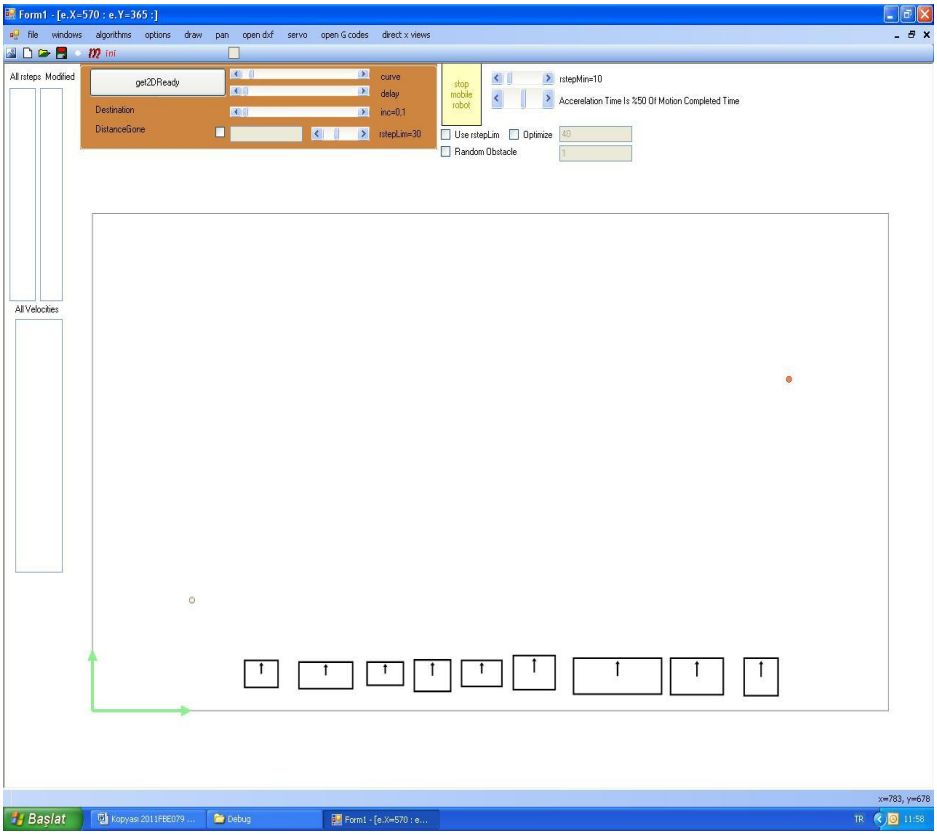
Şekil 2.13 : Hareketli engellerin geçilmesi



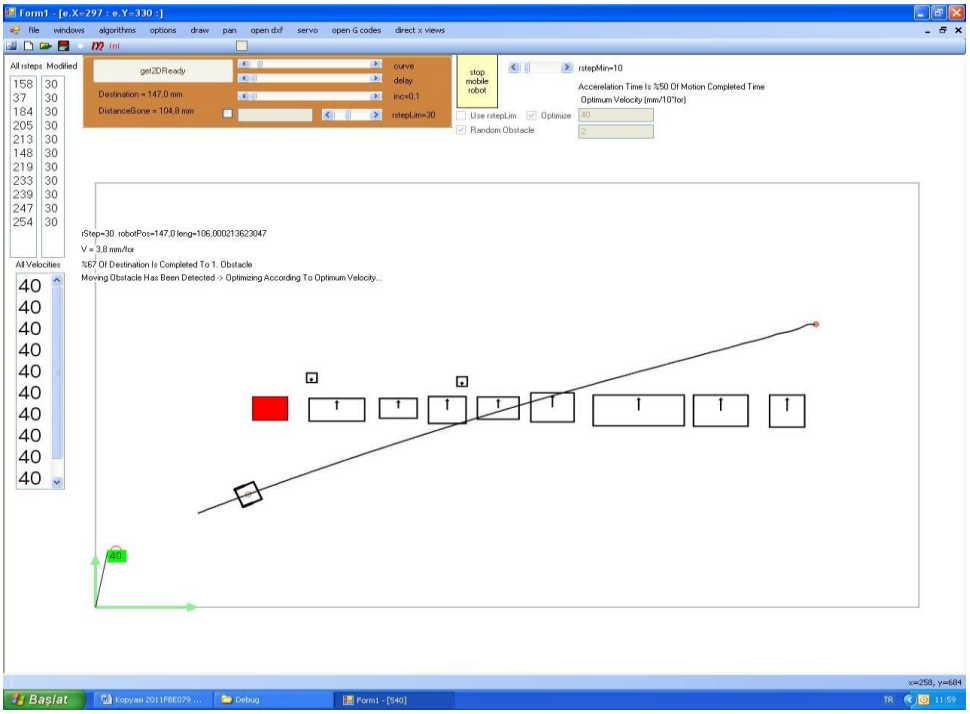
Şekil 2.14 : Mobil robotun hız-zaman grafiği

2.6.3 Sabit, hareketli ve bilinmeyen engellerin geçilmesi

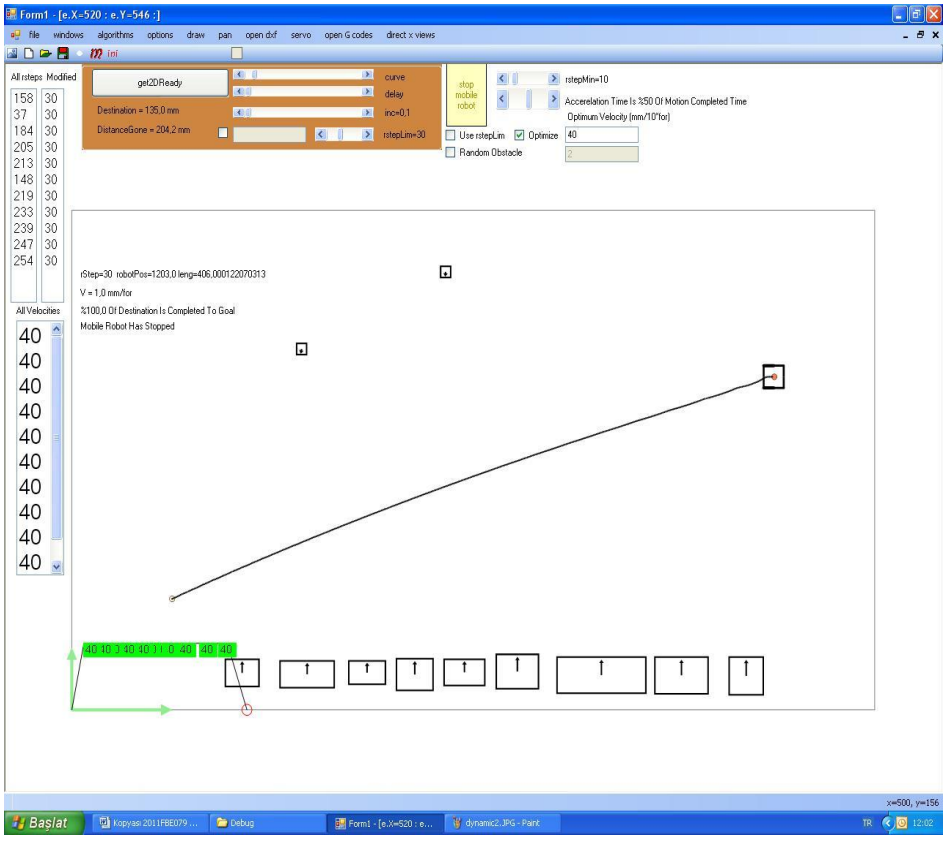
Yazılımda sabit, hareketli ve bilinmeyen engellerin olduğu Şekil 2.15'de görüldüğü gibi hareketli bir ortam hazırlanmıştır. Bilinmeyen engeli açıklamak gerekirse konumu, hareket doğrultusu ve hızı bilgisayar tarafından verilen rastgele bir engeldir ve mobil robot harekete başladıktan sonra ekranda belirmektedir. İki tane rastgele engel 'Random Obstacle' isimli checkbox işaretlenerek mobil robot hareketine başladıktan sonra ekranda görünmeye başlamıştır (Şekil 2.16). Engellerin birbirine çarpması ihmal edilmiş olup birbirlerinin içerisinden hareketine izin verilmişir. Çünkü daha önce de belirtildiği gibi engellerin yükseklik ve genişlik değerleri ihmal edilmektedir. Bu hareketli ortamda mobil robot engellere çarpmadan hedefine ulaşmıştır (Şekil 2.17).



Şekil 2.15 : Hareketli ortamın hazırlanması



Şekil 2.16 : Rastgele engellerin ekranda belirmesi



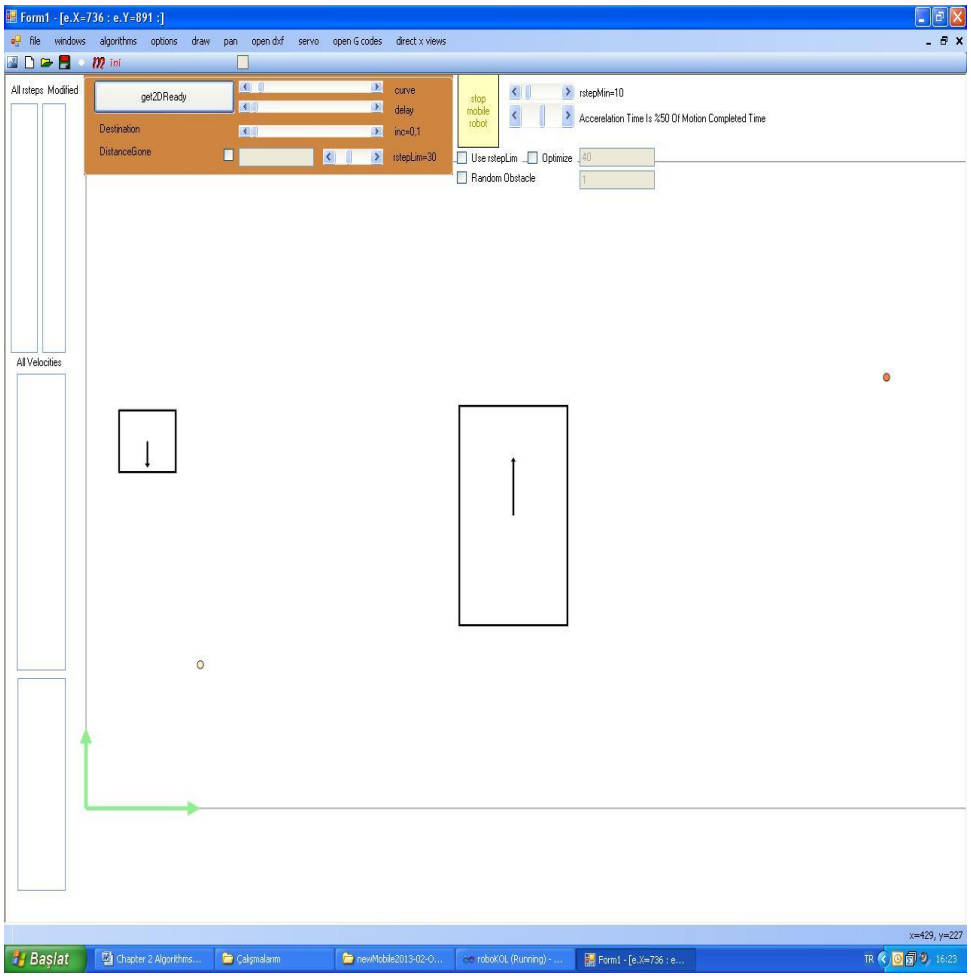
Şekil 2.17 : Mobil robotun hedefe varması

2.7 Yazılımda Çarpma Riski Taşımayan Engellerin Tespit Edilmesi

Yazılımda mobil robota çarpma riski taşımayan engellerin tespiti ve bu engellerin mobil robotun hareketinde görmezden gelinmesi üzerine çalışılmıştır. Ayrıca tasarlanan mobil robotun yazılımda hesaplanan ve animasyon olarak gösterilen hareketi yapabilmesi için mobil robotun sol ve sağ tekerlerinin ne kadar hızla dönmesi gerektiğinin hesabı üzerine çalışılmıştır.

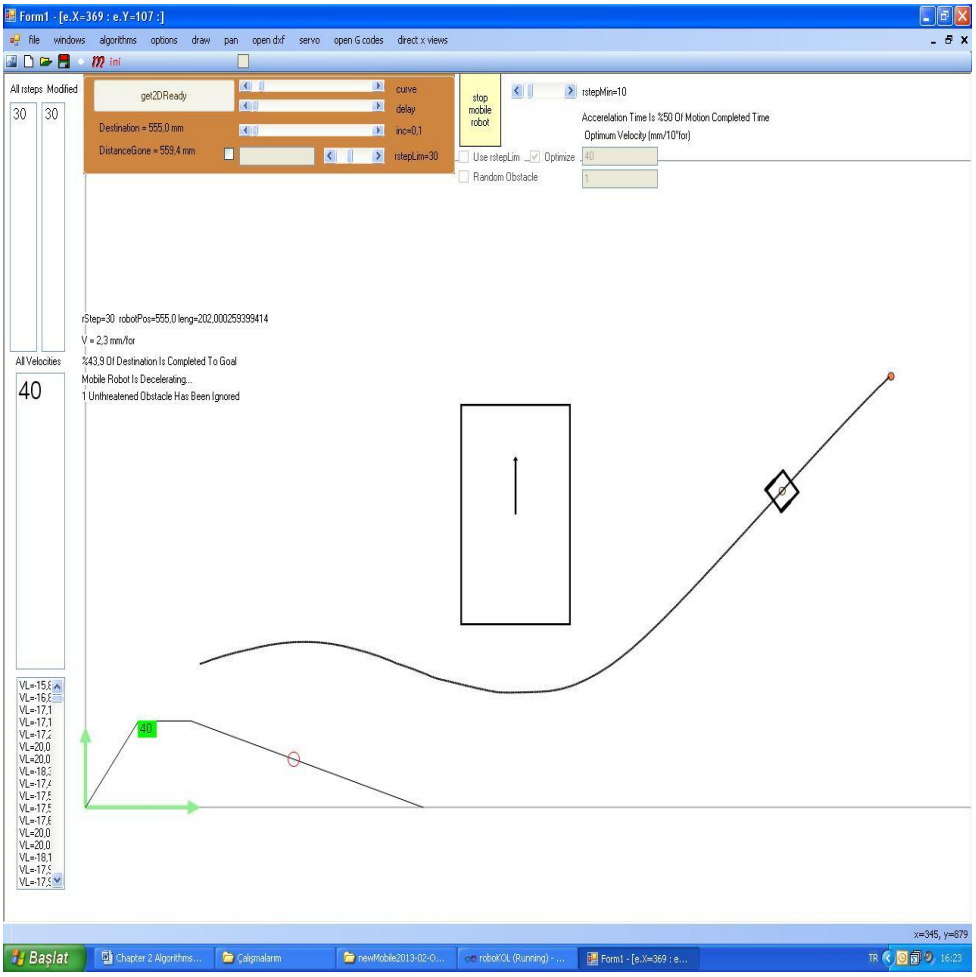
2.7.1 Tehdit oluşturmayan engelin tespiti

Şekil 2.18'deki mobil robotun hareket edeceği ortam incelenecek olursa bu ortamda toplam iki tane engel bulunmaktadır. Birinci engelin hareket doğrultusu başlangıç noktasının arkasına doğru olduğu için bu engel mobil robota çarpma riski teşkil etmemektedir.



Şekil 2.18 : Mobil robota çarpma riski taşımayan engel

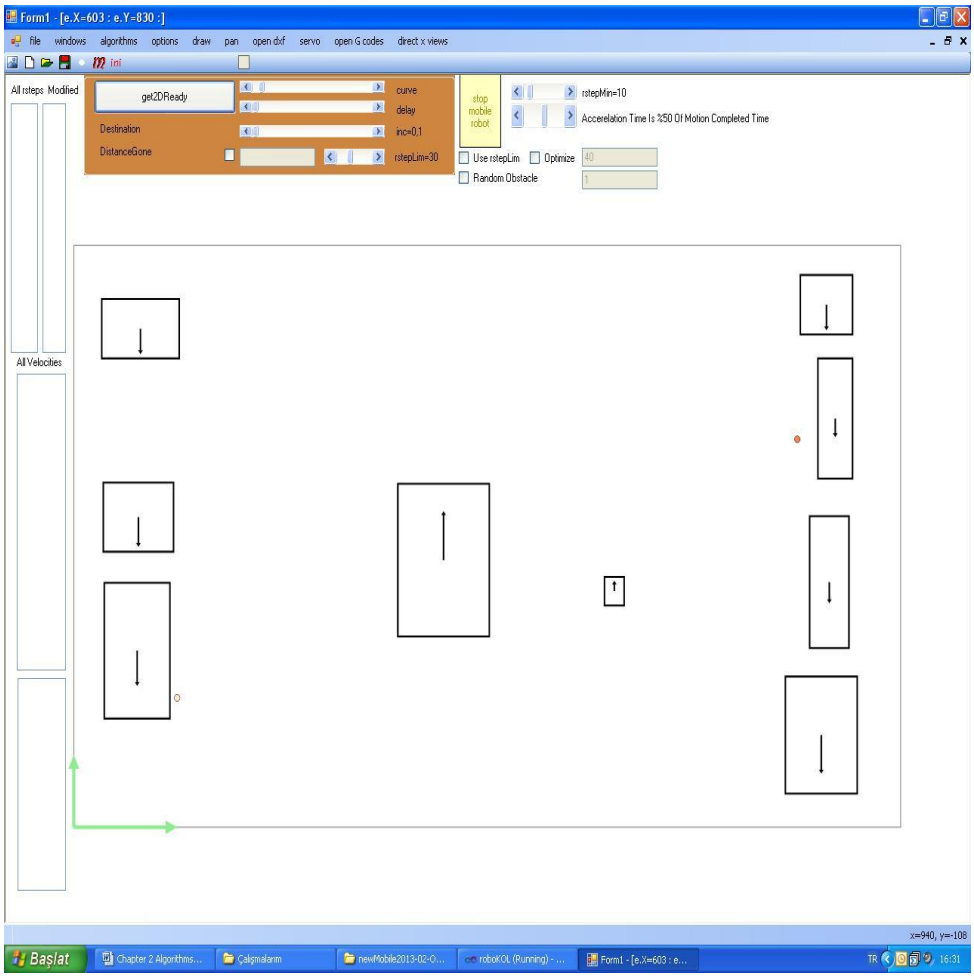
İkinci engel ise sabit bir engeldir. Bu yazılımda birinci engel mobil robot hareketine başladığında mobil robota çarpma riski taşımadığı için görmezden gelinmiştir ve mobil robot sadece sabit engeli geçerek hedefine ulaşmıştır (Şekil 2.19).



Şekil 2.19 : Mobil robotun engeli görmezden gelerek hedefine ulaşması

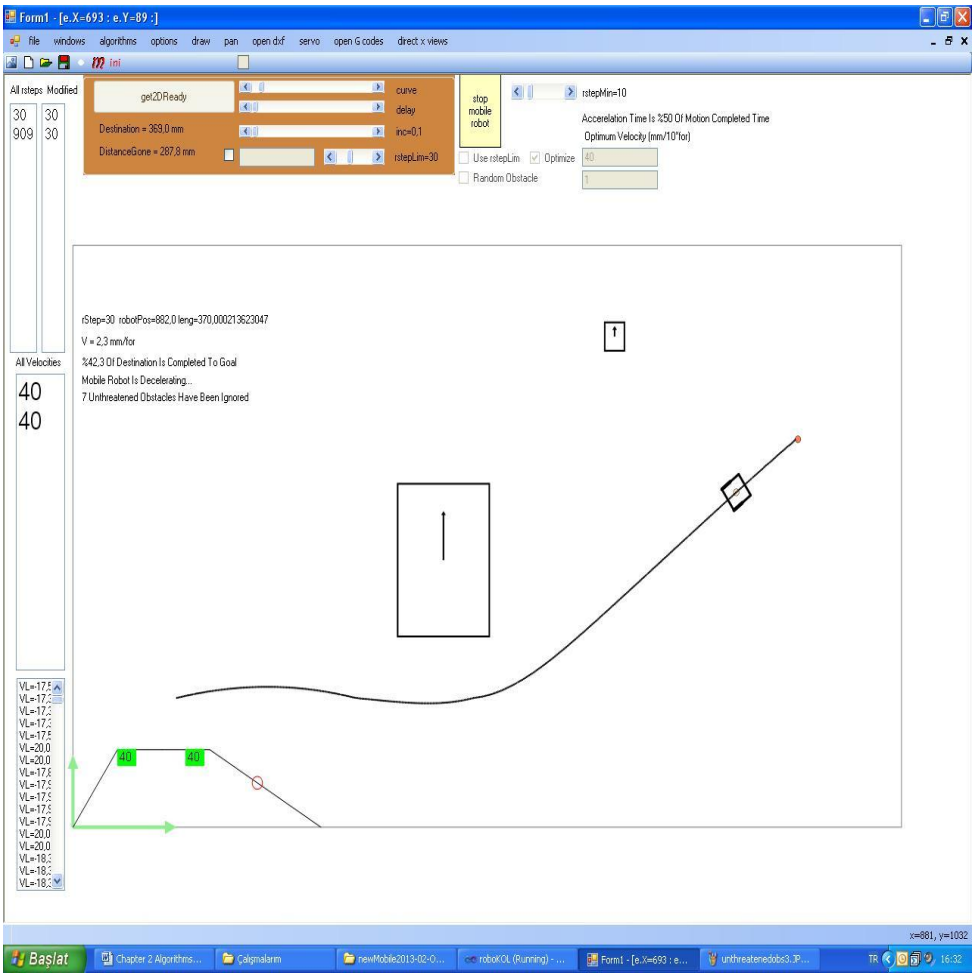
2.7.2 Tehdit oluşturmayan 7 engelin tespiti

Şekil 2.20'deki mobil robotun hareket edeceği ortamda toplam dokuz tane engel bulunmaktadır. Başlangıç noktasının yanındaki üç ve hedef noktasının yanındaki dört engelin mobil robota çarpma riski yoktur. Bu yedi engel haricinde kalan 2 engelden birincisi sabit diğeri ise hareketli bir engeldir.



Şekil 2.20 : Mobil robota çarpma riski taşımayan yedi adet engel

Mobil robot hareketine başladığında tehdit oluşturmayan bu yedi adet engel tespit edilerek görmezden gelinmiştir. Mobil robot sabit ve hareketli olan iki engeli geçerek hedefine ulaşmıştır (Şekil 2.21).



Şekil 2.21 : Mobil robotun yedi adet engeli görmezden gelerek hedefine ulaşması

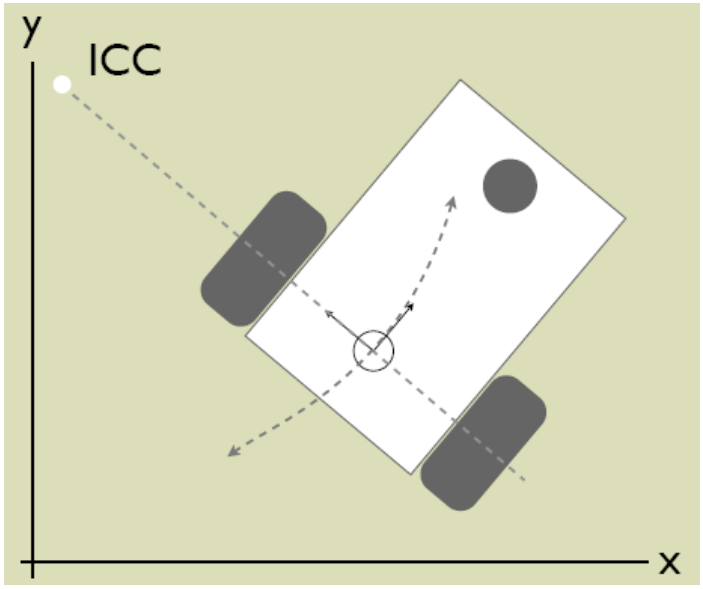
2.7.3 Diferansiyel sürüş sistemi

Mobil robotta sürüş sistemi olarak diferansiyel sürüş sistemi kullanılmıştır. Diferansiyel sürüş sisteminin kullanılmasından dolayı mobil robotun sol ve sağ tekerleri farklı hızlarda döndürülebilmektedir. Bu durum mobil robotun manevra yeteneğini oldukça artırmaktadır. Mobil robotun hareketi esnasında dengesini koruyabilmesi için ilaveten avare tekerler de kullanılmıştır. 2.28 denkleminde V_f doğrusal hızı, ϕ_l sol teker hızı, ϕ_r sağ teker hızı, u tekerlek yarıçapını, 2.29 denklemindeki θ açısal hızı, l aks uzunluğunu ve 2.30 denklemindeki R_{ICC} ise aks merkezi ile mobil robotun etrafında döndüğü nokta arasındaki uzaklığı simgelemektedir. Mobil robotun etrafında döndüğü nokta [14] Şekil 2.22'de gösterilmiştir.

$$V_f = \frac{u}{2}(\varphi_r + \varphi_l) \tag{2.28}$$

$$\theta = \frac{u}{l}(\varphi_r - \varphi_l) \tag{2.29}$$

$$R_{ICC} = \frac{l}{2}\frac{\phi_r + \phi_l}{\phi_r - \phi_l} \tag{2.30}$$

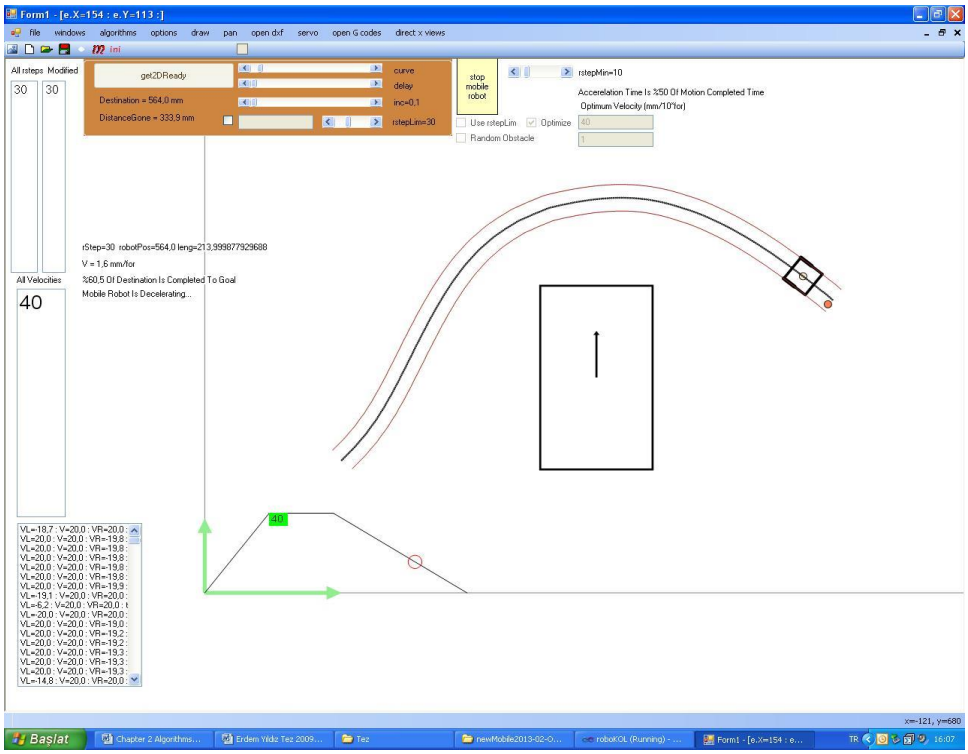


Şekil 2.22 : Mobil robotun dönme merkezi

Yukarıda diferansiyel sürüş sisteminin mobil robotun manevra yeteneğini arttırdığından bahsedilmişti. Bu durum 2.28, 2.29 ve 2.30 denklemleri yardımıyla örneklendirilebilir. Mobil robotun sol ve sağ tekerinin aynı hızla fakat farklı yönlerde döndüğü durum incelenirse ($\phi_r = -\phi_l$), 2.28 denklemindeki doğrusal hız sıfıra eşit olmaktadır ($V_f = 0$). 2.29 denklemindeki θ açısal hızı alabileceği en büyük değeri almaktadır. Ayrıca 2.30 denklemindeki R_{ICC} mobil robotun dönme merkezine uzaklığı sıfıra eşit olmaktadır. Bu durum mobil robotun kendi ekseni etrafında döndüğünü açıklamaktadır. Sol ve sağ tekerin yine aynı hızda fakat bu sefer aynı yönde döndükleri durum incelenecek olunursa ($\phi_r = \phi_l$), θ açısal hızı sıfıra eşit olmaktadır ($\theta = 0$). Bu durum mobil robotun düz bir şekilde hareket ettiğini ifade etmektedir. Ayrıca bu durumda R_{ICC} sonsuz değerini almaktadır ($R_{ICC} = \infty$). Bu sonuç mobil robotun herhangi bir nokta etrafında dönmediğini belirtmektedir.

2.7.4 Diferansiyel sürüş sisteminin yazılımda kullanılması

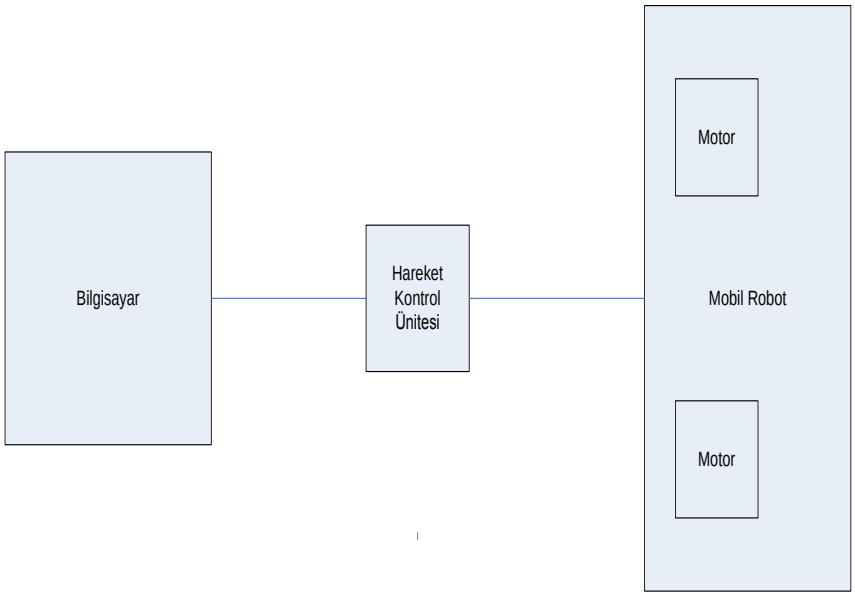
Daha önce işaret edildiği gibi yazılımda mobil robotun engellerden kaçınarak hedefine ulaşması için gereken hızlar hesaplanmaktadır ve bu hız değişimleri grafik olarak çizilmektedir. Mobil robot engellerden kaçınarak hedefine ulaşmaya çalıştığı için manevralar yaparak hedefine ulaşabilmektedir. Dolayısıyla tasarlanan mobil robotun yazılımda belirlenen yolu izleyebilmesi için manevra yapma kabiliyetinin olması gerekmektedir. Diferansiyel sürüş sisteminde sol ve sağ tekerin aynı hızda fakat farklı yönlerde döndüğü durum daha önce incelenerek mobil robotun kendi ekseni etrafında dönebilme kabiliyetinin olduğu matematiksel ifadelerle belirtilmişti. Bu yüzden mobil robot diferansiyel sürüş sistemine sahip olmasından dolayı yüksek bir manevra kabiliyetine sahiptir. Mobil robotun yazılımdaki yolu takip edebilmesi için o yol parçalara ayrılarak sol ve sağ tekerin hızları her parça için hesaplanmaktadır. Şekil 2.23'de görüldüğü gibi sol alttaki liste kutusunda bu hesaplamalar liste halinde kullanıcıya gösterilmektedir. Bu listedeki VL sol tekerin hızı, VR sağ tekerin hızı, V ise bileşke hız değeridir. Bu hız değerleri mobil robotun istenilen manevrayı yapabilmesi için sol ve sağ tekerine redüktör ile bağlı olan iki ayrı servo motorun ulaşması gereken hızlardır.



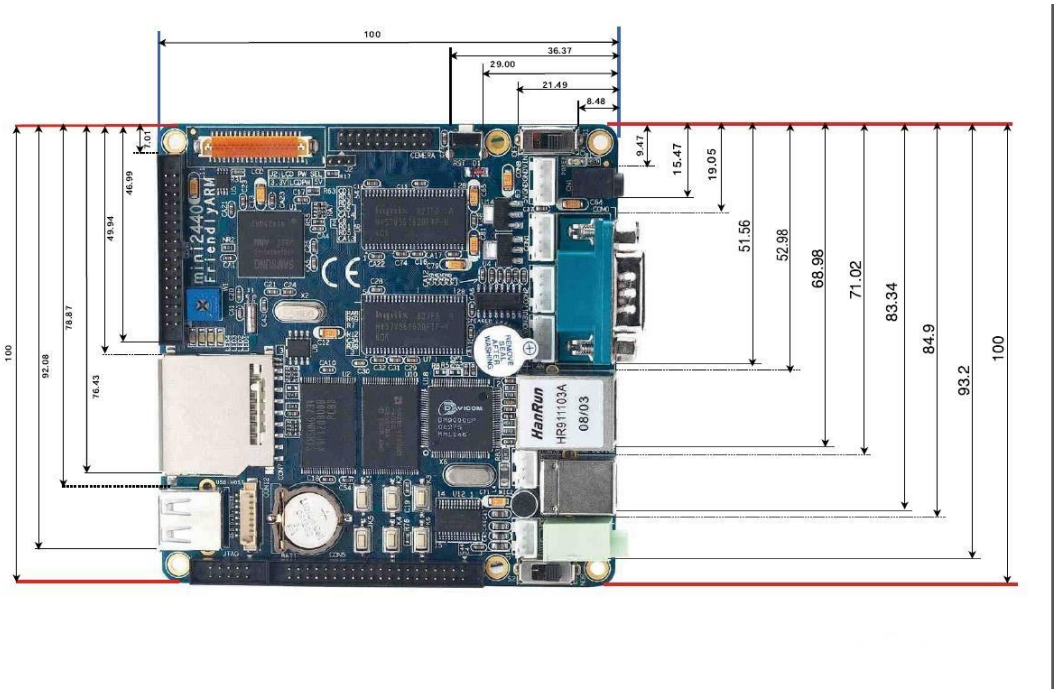
Şekil 2.23 : Mobil robotun sol ve sağ tekerlerinin dönme hızlarının hesabı

3. MINI 2440 GÖMÜLÜ SİSTEM

Üzerinde çalışılan mobil robotun daha önceki tasarımında mobil robotun hareketi Şekil 3.1'de görüldüğü gibi, üzerine monte edilmiş bir bilgisayarın hareket kontrol ünitesine emir göndermesi ve bu emirin de hareket kontrol ünitesi tarafından mobil robotun tekerlerine bağlı olan motorların sürücülerine iletilmesiyle gerçekleştirilmişti. Fakat bilgisayar mobil robot üzerinde çok yer kaplamaktadır. Bu tezde mobil robotun hareketi Şekil 3.2'de görüldüğü üzere 100x100 mm boyutlarında bilgisayardan çok daha az yer kaplayacak olan mini 2440 gömülü sistem kullanılarak gerçekleştirilmiştir.

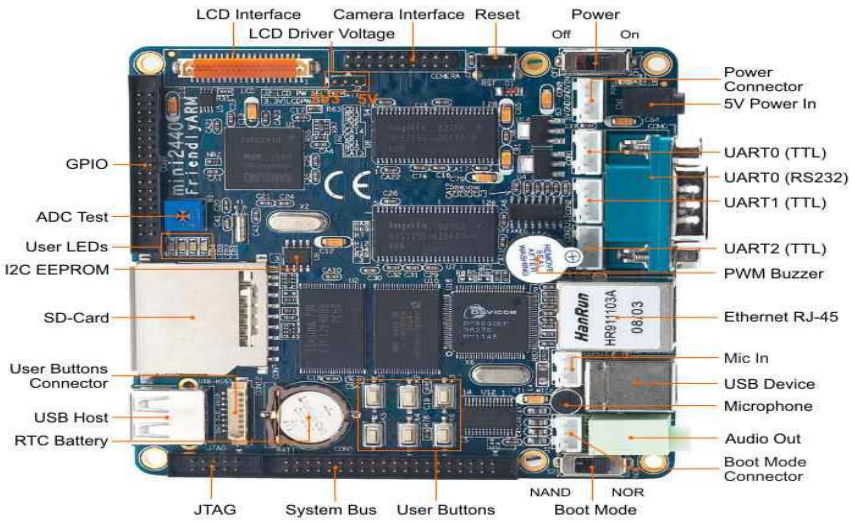


Şekil 3.1 : Mobil robot hareket şeması



Şekil 3.2 : Mini 2440 gömülü sistemin boyutları

Mini 2440 gömülü sistem, boyutu küçük olmasına rağmen üzerinde bilgisayarda bulunan birçok donanımsal elemana sahiptir. SD kart, usb, ethernet girişi, ses çıkışı ve mikrofon girişi gibi önemli donanımlar üzerinde mevcuttur (Şekil 3.3). İşlemci olarak 400 Mhz hızında Samsung S3C2440A işlemcisini kullanmakta ve 64 MB SD RAM kapasitesine sahiptir. Mini 2440 gömülü sistemin Şekil 3.3'de görüldüğü gibi LCD Interface isimli lcd ekran girişi vardır. Mini 2440 ile beraber satın alınan 7 inçlik lcd dokunmatik ekran üzerinde dokunmatik kalem yardımıyla Windows CE işletim sisteminin özellikleri kullanılabilmektedir.



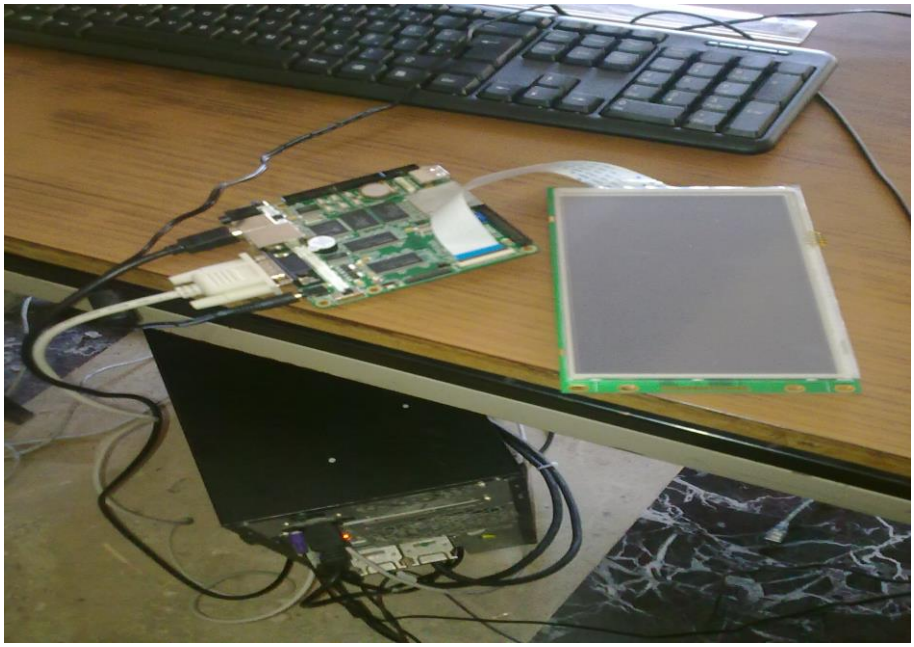
Şekil 3.3 : Mini 2440 gömülü sistemin donanımsal özellikleri

3.1 Mini 2440 Gömülü Sistemin Çalıştırılması

Mini 2440 isimli gömülü sistem elimize içerisine Windows CE işletim sistemi yüklenmiş bir vaziyette ulaşmıştır. Kullanıldıkça yüklü olan işletim sisteminde bazı eksikliklerin olduğu tespit edilmiştir. Bu eksikliklerden en önemlisi masaüstü arka plan resminin değiştirildikten sonra Mini 2440 gömülü sistemin kapatılıp yeniden açıldığında yapılan arka plan resim değişikliğinin kaydedilmemiş olmasıydı. Mini 2440 gömülü sistemin özellikleri kullanıldıkça işletim sistemindeki sürücü dosyalarının eksik olduğu fark edilmiştir ve satıcı firma ile görüşülmüştür. Satıcı firma tarafından üretici firmanın web sitesinin [15] araştırılması gerektiği tarafımıza iletilmiştir ve o web sitesi araştırıldıktan sonra orada Windows CE işletim sistemi ve sürücüleri içeren bir imaj dosyası olduğu görülmüştür. O web sitesinden ilgili dosyalar indirilmiştir ve bu dosyalar yardımıyla Mini 2440 gömülü sisteme Windows CE işletim sistemi yeniden yüklenmiştir.

3.2 Mini 2440 Gömülü Sisteme Windows CE İşletim Sisteminin Kurulması

Öncelikle masaüstü bilgisayara usb download driver yüklenerek Mini 2440 gömülü sistem bu bilgisayara tanıtılmıştır. Daha sonra bilgisayar ile Mini 2440 gömülü sistem, usb ve seri port kablosu ile birbirine bağlanmıştır (Şekil 3.4).



Şekil 3.4 : Mini 2440 gömülü sistemin usb ve seri port kabloları ile bilgisayara bağlanması

Şekil 3.5'de görüldüğü gibi boot switch NOR kısmına getirildikten sonra cihaz kapalı durumda tutulmuştur.

Şekil 3.5 : Boot switch elemanının NOR kısmına getirilmesi

Bilgisayarın COM port numarası tespit edildikten sonra bilgisayarda, üretici firmanın web sitesinden [15] indirilen seri port bağlantısını sağlayan DNW programı çalıştırılmıştır ve tespit edilen COM port numarası DNW programında seçilerek Mini 2440 gömülü bilgisayar ile bilgisayar arasındaki bağlantı sağlanmıştır. Daha sonra Mini 2440 gömülü bilgisayar çalıştırıldığında bilgisayar ekranındaki DNW programının ekranında Şekil 3.6'da görüldüğü gibi işlemler menüsünün görüntülendiği görülmüştür. İndirilmiş olan ve bilgisayarda bulunan, format için gerekli olan dosyaların gömülü bilgisayara iletilebilmesi için DNW programı kullanılarak bilgisayar ile Mini 2440 gömülü sistem arasında usb bağlantısı sağlanmıştır.

Şekil 3.6 : DNW programı işlem menüsü

Köşeli parantezler arasında belirtilen harflere klavyeden basılarak işlem menüsünde işlem yapılabilmektedir. Öncelikle var olan Windows CE işletim sistemi f tuşuna basılarak formatlanmıştır. Bu formatlama işleminden sonra v tuşuna basılarak supervivi-128M dosyası, n tuşuna basılarak 7 inç ekran için nboot_A70.bin dosyası, l tuşuna basılarak bootlogo.bmp dosyası ve son olarak w tuşuna basılarak usb port aracılığı ile Windows CE işletim sisteminin imajı gömülü bilgisayara yüklenmiştir. Yükleme tamamlandıktan sonra Şekil 3.5'de görüldüğü gibi NOR kısmına getirilen boot switch tekrardan NAND kısmına getirilerek cihaz açılmıştır ve Windows CE işletim sisteminin ekrana geldiği gözlenmiştir. Bu kurulumdan sonra değiştirilen arka plan resmini kaydetmeme ve bazı özelliklerin çalışmaması gibi sorunların çözüldüğü gözlemlenmiştir.

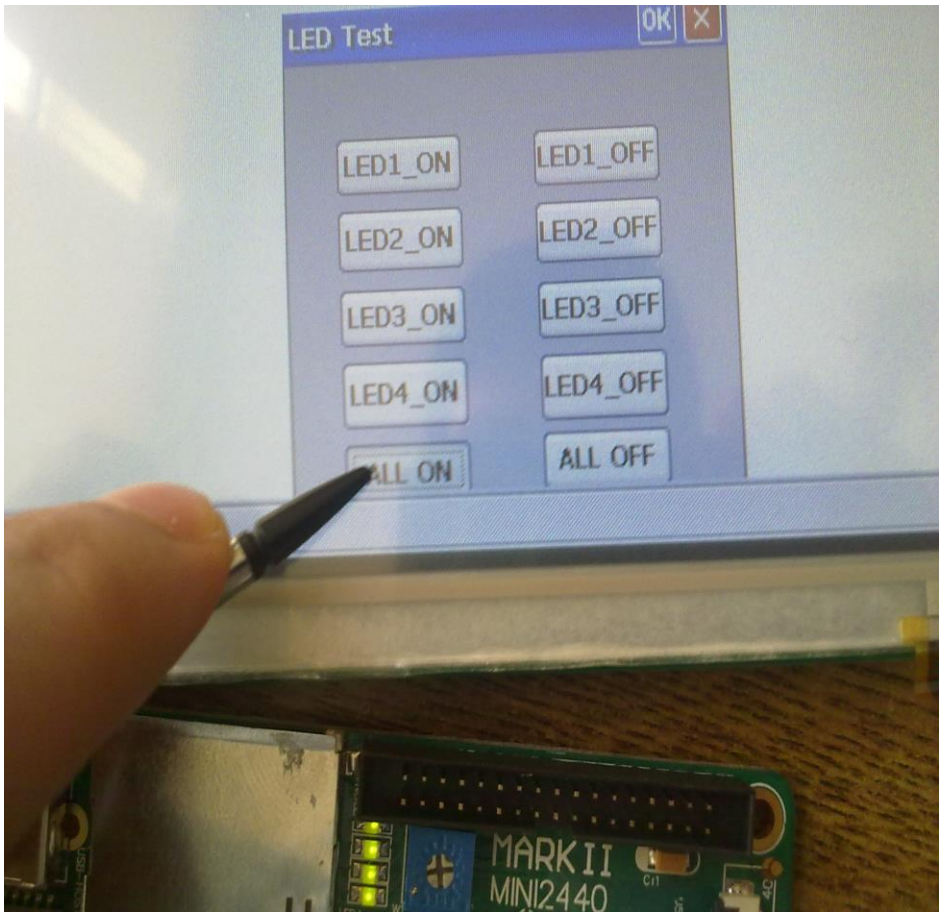
3.3 Mini 2440 Gömülü Sistemdeki Örnek Uygulamalar

Öncelikle Mini 2440 gömülü sistemin özelliklerinin çalışıp çalışmadığının anlaşılabilmesi için test amaçlı olarak, yüklenmiş olan imaj dosyasında bulunan Mini 2440 uygulamaları çalıştırılmıştır. Daha sonra Microsoft Visual Studio NET programı kullanılarak örnek programlar geliştirilmiştir.

3.3.1 Test amaçlı olan örnek uygulamalar

3.3.1.1 Led uygulaması

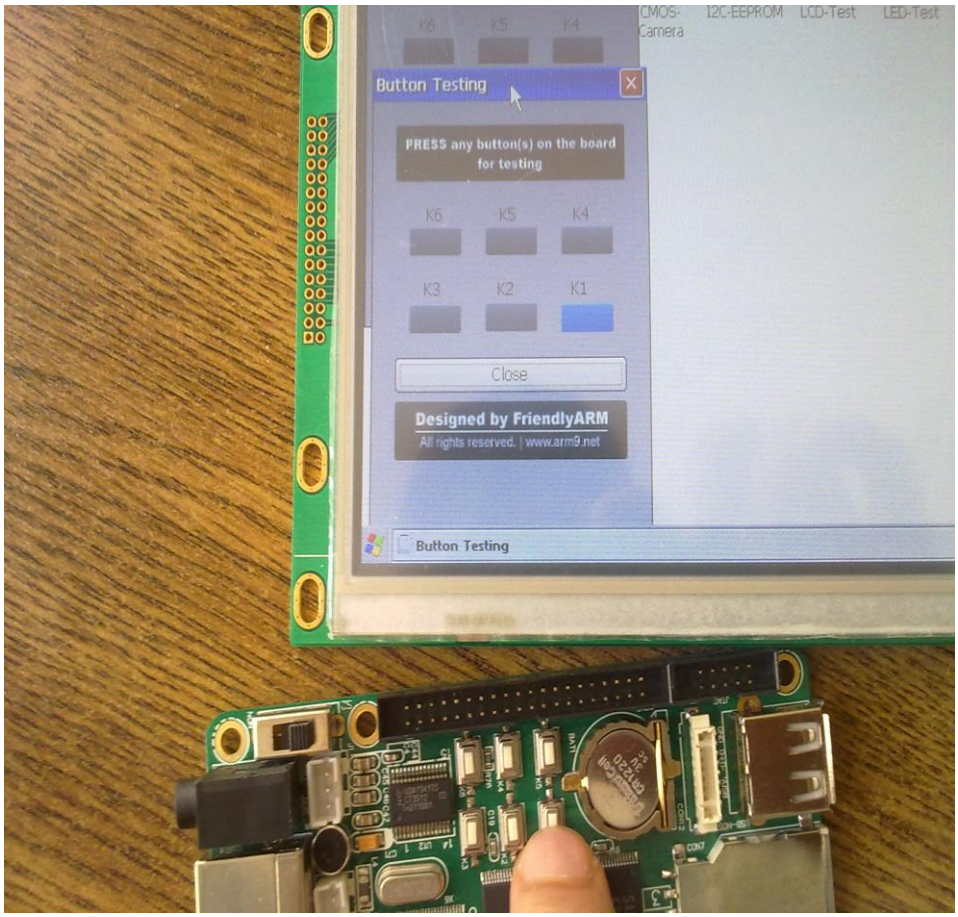
Mini 2440 gömülü bilgisayar üzerinde bulunan dört adet led ışıklarını yakıp söndürmeye yarayan test amaçlı olarak yazılmış üretici firmanın web sitesine [15] koyulmuş olan bir uygulamadır. Şekil 3.7'de görüldüğü gibi ALL ON isimli düğmeye tıklanarak bütün ledler yakılmıştır.



Şekil 3.7 : Led uygulaması

3.3.1.2 Düğme uygulaması

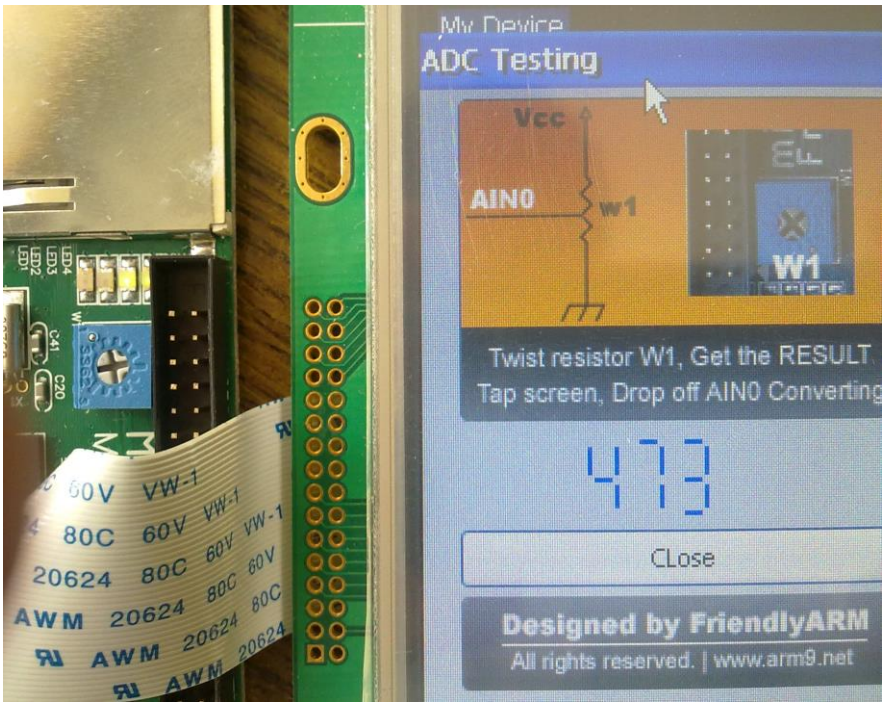
Mini 2440 gömülü bilgisayar üzerinde bulunan altı adet düğmenin her birine basılıp basılmadığını kontrol eden test amaçlı olarak yazılmış üretici firmanın web sitesine koyulmuş olan bir uygulamadır (Şekil 3.8).



Şekil 3.8 : Düğme uygulaması

3.3.1.3 Adc uygulaması

Şekil 3.9'da görüldüğü gibi Mini 2440 gömülü bilgisayar üzerinde bulunan direncin değerine göre çıkan analog sinyalin değerini dijitale dönüştüren ve bu değeri ekranda yazan test amaçlı olarak yazılmış üretici firmanın web sitesine koyulmuş olan bir uygulamadır.

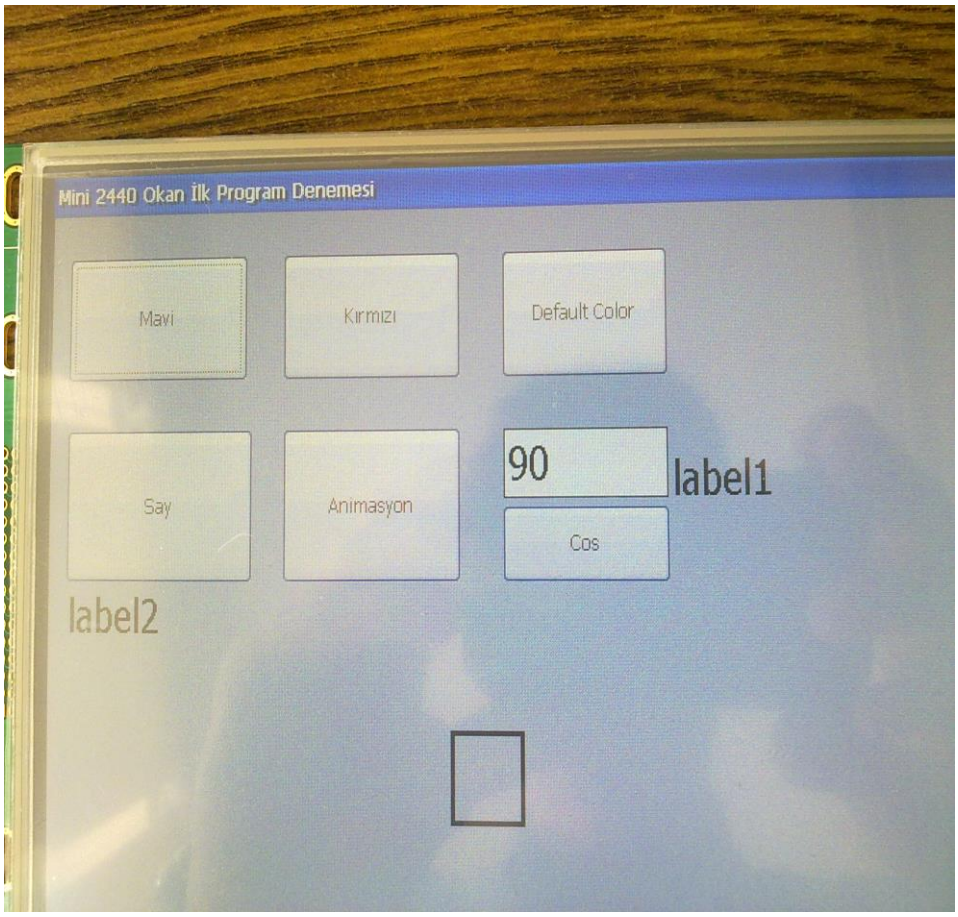


Şekil 3.9 : Adc uygulaması

3.3.2 Microsoft Visual Studio NET Kullanılarak Geliştirilen Uygulamalar

3.3.2.1 Deneme amaçlı geliştirilen uygulama

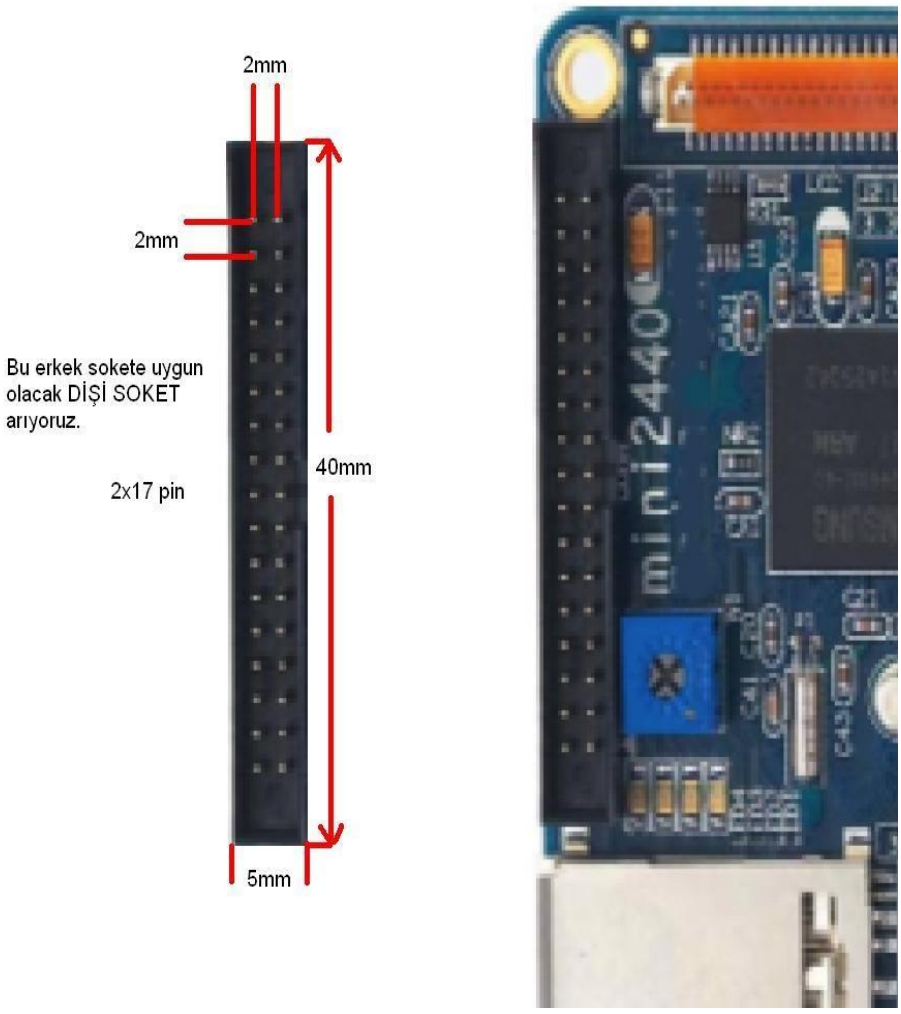
Microsoft Visual Studio 2008 programı kullanılarak Mini 2440 Gömülü sistemde çalıştırılmak üzere uygulamalar geliştirilmiştir. İlk olarak Windows CE işletim sisteminde programlamada çok kullanılan fonksiyonların nasıl çalıştığının gözlenmesi amacıyla Şekil 3.10'da görülen "Mini 2440 Okan İlk Program Denemesi" isimli yazılım geliştirilmiştir. Bu yazılımda formun arka planı değiştirilerek Windows CE işletim sisteminde renklerin nasıl göründüğü, say düğmesi yardımıyla for döngüsünün kaç kere döndürülebildiği, trigonometrik işlemlerdeki doğruluk payı ve son olarak karenin hareket animasyonunun nasıl çalıştığı gözlenmiştir. Sonuç olarak formun arka plan renk görünümünde bir düzensizliğe rastlanılmamıştır. Ayrıca trigonometrik işlemin sonucunun doğru çıktığı gözlenmiştir. For döngüsünün çalışmasında ise Windows XP veya Windows 7 işletim sistemini çalıştıran bilgisayarlarla fark olmadığı tespit edilmiştir. Olumsuz anlamda sadece kare animasyonunda titreme sorununun olduğu gözlenmiş ve bu sorunu giderebilecek kod yapısının Windows CE işletim sistemi için geliştirilecek proje tipinin kütüphanesinde bulunmadığı anlaşılmıştır.



Şekil 3.10 : Mini 2440 gömülü sistem için geliştirilen ilk yazılım

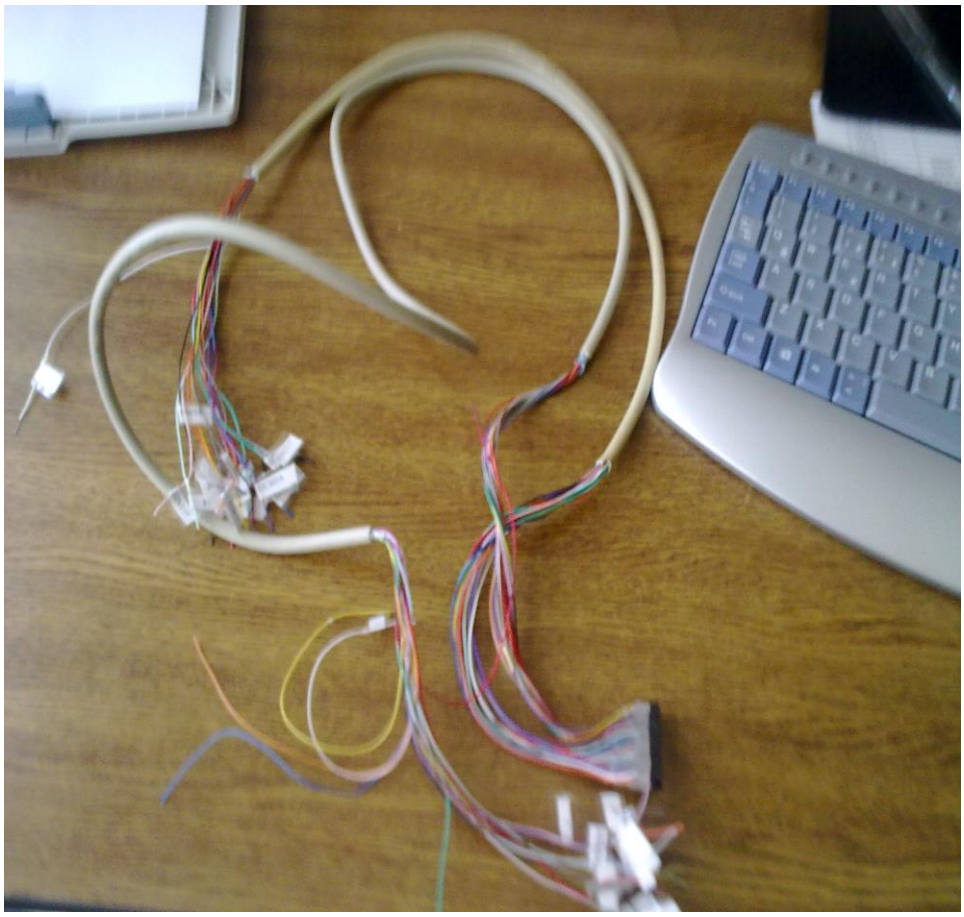
3.3.2.2 Pulse üretilerek step motoru sürme amacıyla geliştirilen yazılım

Mini 2440 gömülü sistem kullanılarak pulse üretimi bu tezde büyük önem arz etmektedir. Bu üretilecek pulse ile mobil robotun hareketinde kullanılacak motorların döndürülmesi hedeflenmektedir. Pulse üretimi yapabilmek için donanımsal olarak giriş çıkış portlarına ihtiyaç duyulmaktadır. Mini 2440 gömülü sistemde bulunan GPIO (Genel Amaçlı Giriş Çıkış) portları pulse üretimi için kullanılmıştır (Şekil 3.11).



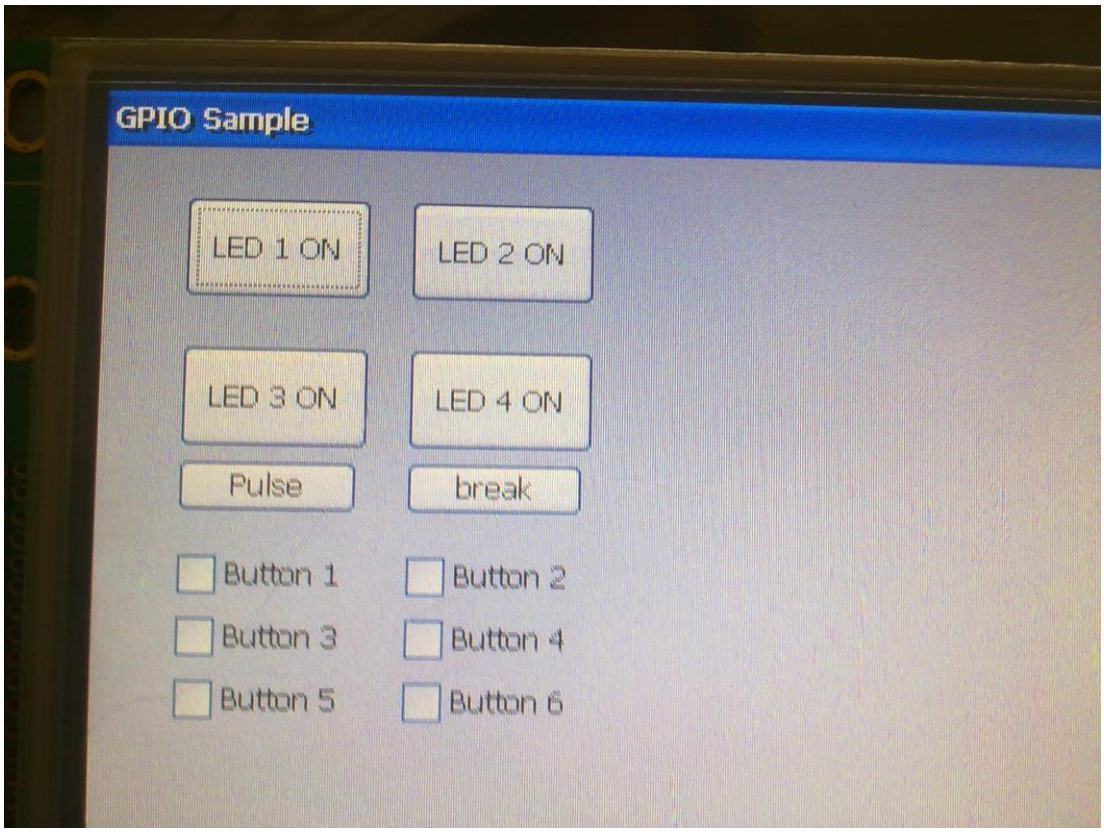
Şekil 3.11 : Mini 2440 gömülü sistem GPIO portları

Mini 2440 cihazın kısa devre yapmaması için GPIO portlarına doğrudan kablo bağlanmaması uygun görülmüştür. Bağlantının bir soket yardımı ile yapılması düşünülmüştür. Bunun için 2 mm çapında 2x17 bir soket gerektiği tespit edilmiştir. Bu soket standartların dışında olduğundan bu soket kolaylıkla bulunamamışır. İnternetten araştırıldığında bu soketin mini 2440 cable kit isminde bir ürün olduğu ve yurtdışında satıldığı görülmüştür. Fiyatı çok ucuz olmasına rağmen siparişin gemi ile getirilmesinden dolayı bu ürünün fiyatının arttığı görülmüştür ve bu nedenle bu ürün satın alınıp kullanılamamıştır. Bu yüzden başka alternatifler düşünülmüştür ve son olarak 2 mm kalınlığında 2 tane 1x40 lık soket çok ucuza satın alındıktan sonra 17 pinlik kısımları yan keski yardımı ile kesilerek bu soketler temin edilmiştir. Daha sonra paralel port kablosunun içindeki kablolar bu soketlerin pinlerine tek tek lehim yapılmıştır ve son olarak bu iki soketin silikon ile birbirine yapıştırılmasıyla Şekil 3.12'de görüldüğü gibi 2x17 lik soket kullanıma hazır hale getirilmiştir.



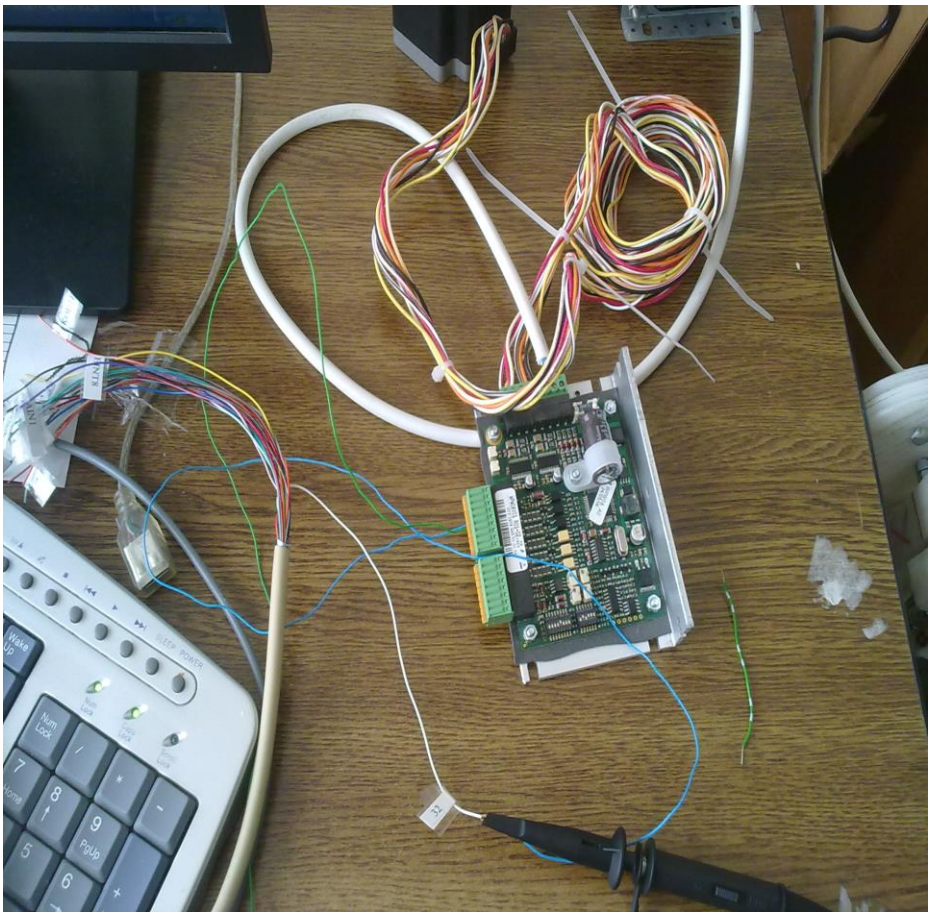
Şekil 3.12 : GPIO portları için 2x17 lik soket

Ayrıca bu portların kullanılarak pulse üretilebilmesi için istenildiğinde ilgili portlara voltaj verebilecek ve voltajı kesebilecek bir yazılım gerekmektedir. Şekil 3.13'de görülen yazılım, deneme amaçlı yazılmış yazılımın üzerine pulse üretmeyi sağlayan Pulse isimli düğme eklenerek oluşturulmuştur. Yazılımda öncelikle satıcı firmadan tedarik edilen GPIO portlarına ulaşılmasını sağlayan kod parçacıkları kullanılmıştır.



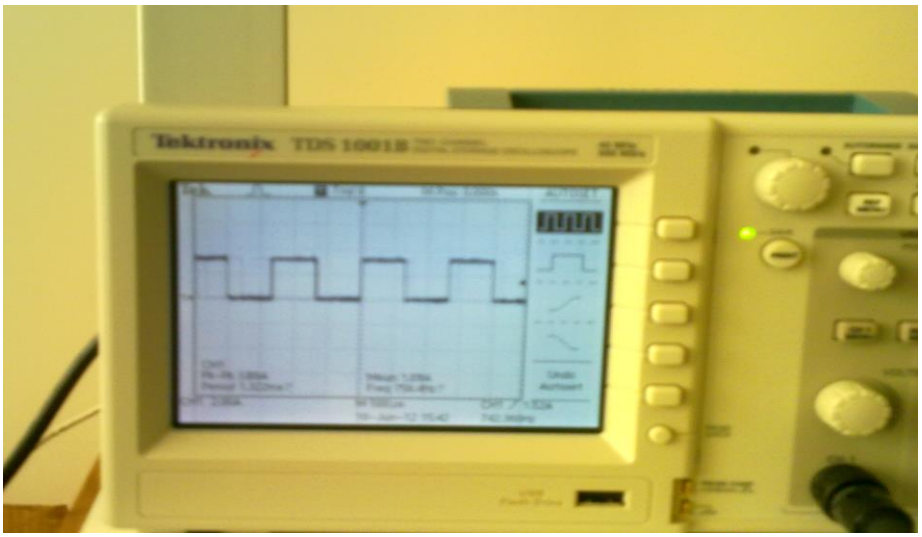
Şekil 3.13 : Pulse üretmek amacıyla geliştirilen yazılım

Sonra B portunun 1. pini Şekil 3.14'de görüldüğü gibi 32. pini yani GPB1 portu çıkış olarak kullanılarak belli aralıklarla voltaj verilmiş ve kesilmiştir.



Şekil 3.14 : GPIO portunun 32. pininin GPB1 portunun çıkış olarak kullanılması

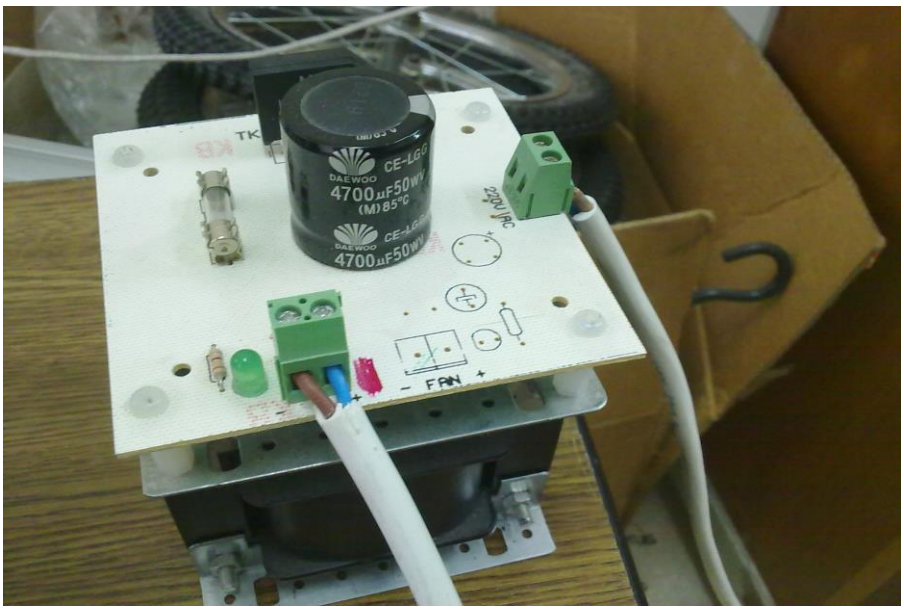
Şekil 3.15'deki osiloskop ekranında görüldüğü gibi kare dalga oluşturularak step motor sürülmüştür.



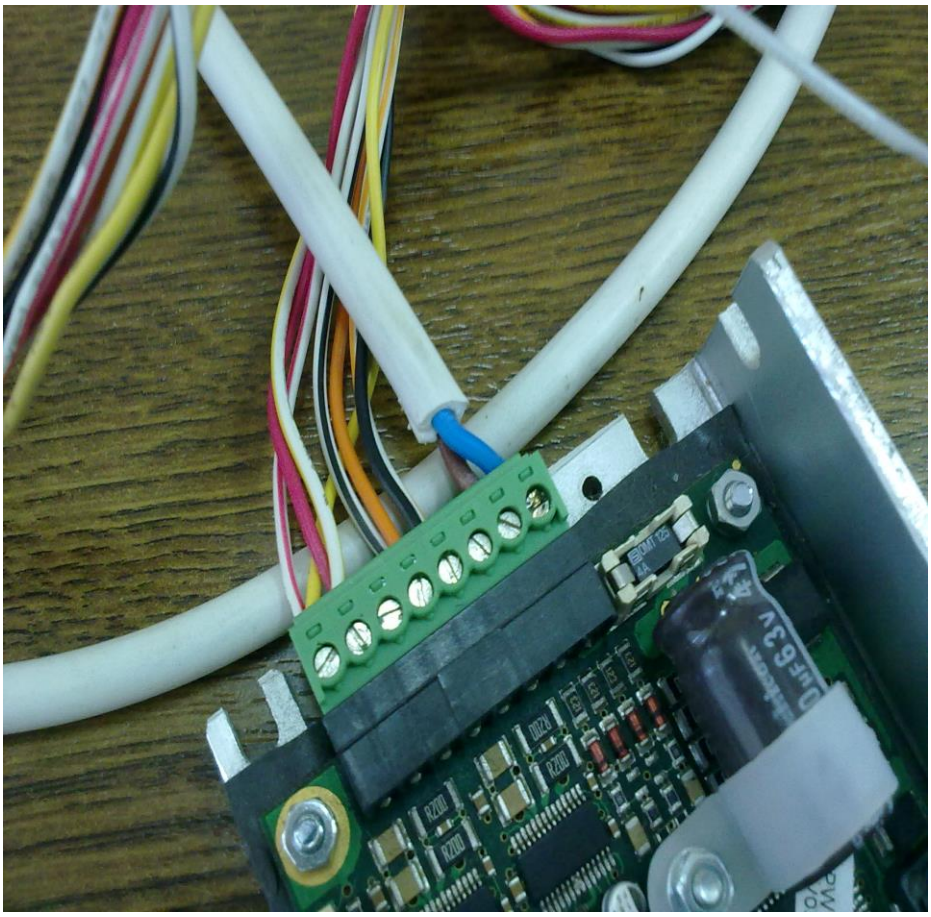
Şekil 3.15 : Kare dalga oluşturulması

Şekil 3.16'de görüldüğü üzere step motor, 40V güç kaynağı (Şekil 3.17) ve step motor sürücü devresi (Şekil 3.18) yardımıyla sürülmüştür.

Şekil 3.16 : Step motorun sürülmesi



Şekil 3.17 : 40V güç kaynağı



Şekil 3.18 : Step motor sürücü devresi

Step motor bu şekilde istenilen hızda sürülememiştir. Bu durum kullanılan yazılımın Visual C# programlama dilinde yazılması ve Mini 2440 gömülü sistemin bir işletim sistemi çalıştırıyor olmasından dolayı gerçek zamanlı olarak pulse'ın motora gönderilememesinden kaynaklanmaktadır. Daha sonraki bölümde de açıklanacağı üzere STM32F4 Discovery isimli işletim sistemi çalıştırmayan yine Mini 2440 gömülü sistemdeki gibi ARM işlemciye sahip bir board yardımıyla bu sorun çözülmüştür.

3.3.3 Haberleşme amaçlı geliştirilen yazılımlar

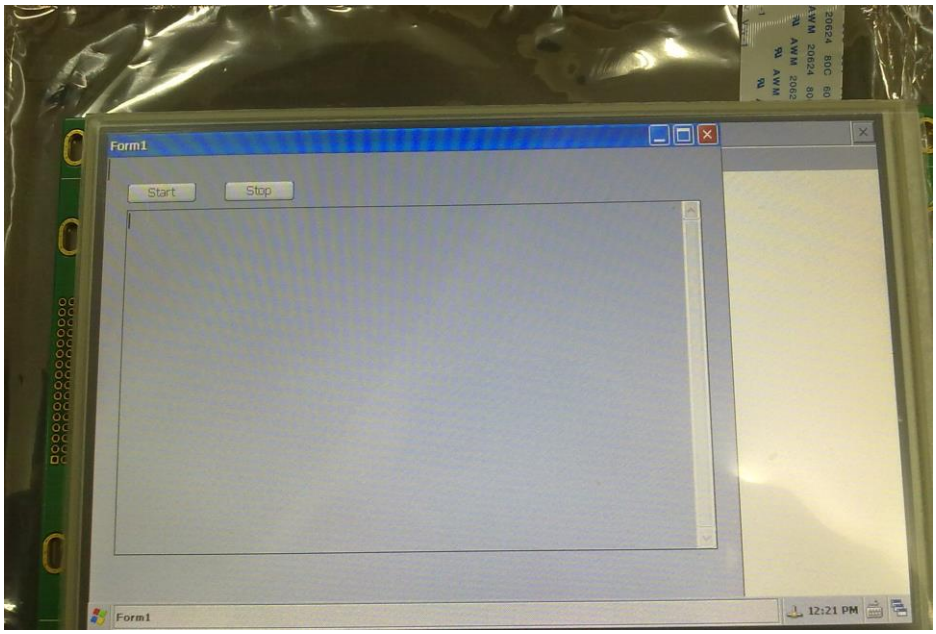
Mini 2440 gömülü sistemin mobil robotun hareketini sağlayan motorların sürücülerine motorların ne kadar dönmesi gerektiği bilgisini gönderebilmesi gerekmektedir. Daha önceki bölümdeki Şekil 2.23'de görüldüğü gibi mobil robot hareket yazılımında diferansiyel sürüş sistemi kullanılarak mobil robotun yazılımda hesaplanan yolu takip edebilmesi için gerekli olan sol ve sağ teker hızları liste olarak kullanıcıya gösterilmiştir. Bu değerlerin mobil robotun hareketini sağlayan motorların sürücülerine aktarılması gerekmektedir. Bu nedenle Mini 2440 gömülü sistem ile haberleşme ihtiyacı duyulmuştur. Mini 2440 gömülü sistemin üzerinde RS–232 ve Ethernet haberleşme protokolü ile haberleşmenin yapılabilmesi için gerekli olan donanımlar mevcuttur. Dolayısıyla öncelikle bu iki farklı haberleşme protokolünün kullanılarak ara aygıt ile haberleşmenin sağlanabilmesi için gerekli olan yazılımlar Microsoft Visual Studio 2008 programı kullanılarak geliştirilmiştir.

3.3.3.1 RS–232 haberleşme protokolü

Mini 2440 gömülü sistemimizde RS–232 haberleşmesi yapabilmek için gerekli donanım bulunmaktadır. Daha önce de belirtildiği üzere Mini 2440 gömülü sisteme Windows CE işletim sistemi bu haberleşme protokolü kullanılarak yüklenmişti (Şekil 3.5). Haberleşmenin gerçekleşebilmesi için haberleşecek iki aygıtta da RS–232 bağlantısı ile karakter gönderebilen ve gönderilen karakteri de algılayabilen yazılımların çalışması gerekmektedir. Dolayısıyla bu haberleşme protokolünün kullanılarak Mini 2440 ile bağlanacak ara aygıtın haberleşmesinin sağlanması amacıyla Şekil 3.19'da görüldüğü gibi Microsoft Visual Studio 2008 programı kullanılarak bir yazılım geliştirilmiştir.

Şekil 3.19 : RS–232 haberleşmesi için geliştirilen yazılım

Ayrıca Mini 2440 gömülü sistemde çalıştırılmak üzere RS–232 haberleşmesinin gerçekleştirilebilmesi için gerekli olan başka bir yazılım geliştirilmiştir (Şekil 3.20).

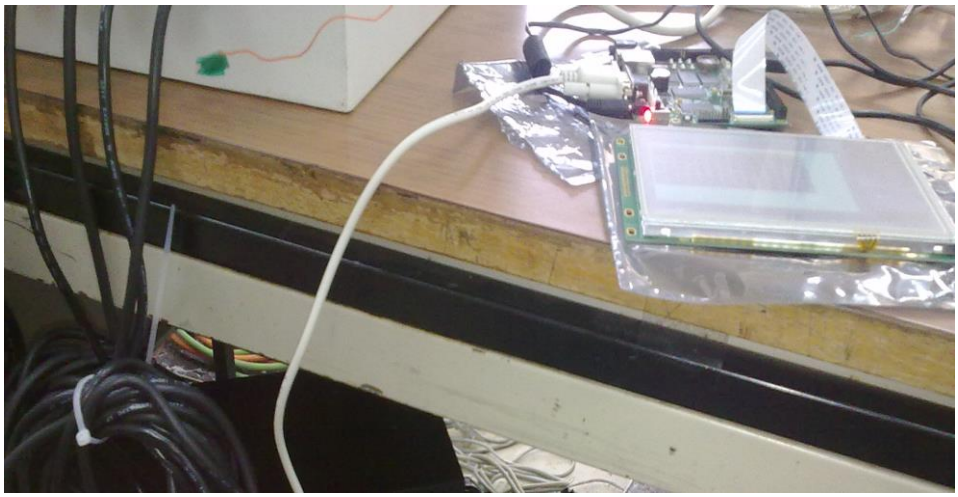


Şekil 3.20 : Mini 2440 gömülü sistemde RS–232 haberleşmesi için geliştirilen yazılım

Haberleştirilecek aygıtlar için gerekli yazılımlar geliştirildikten sonra iki aygıt Şekil 3.21'de görülen RS–232 bağlantı kablosu ile birbirine bağlanmıştır.

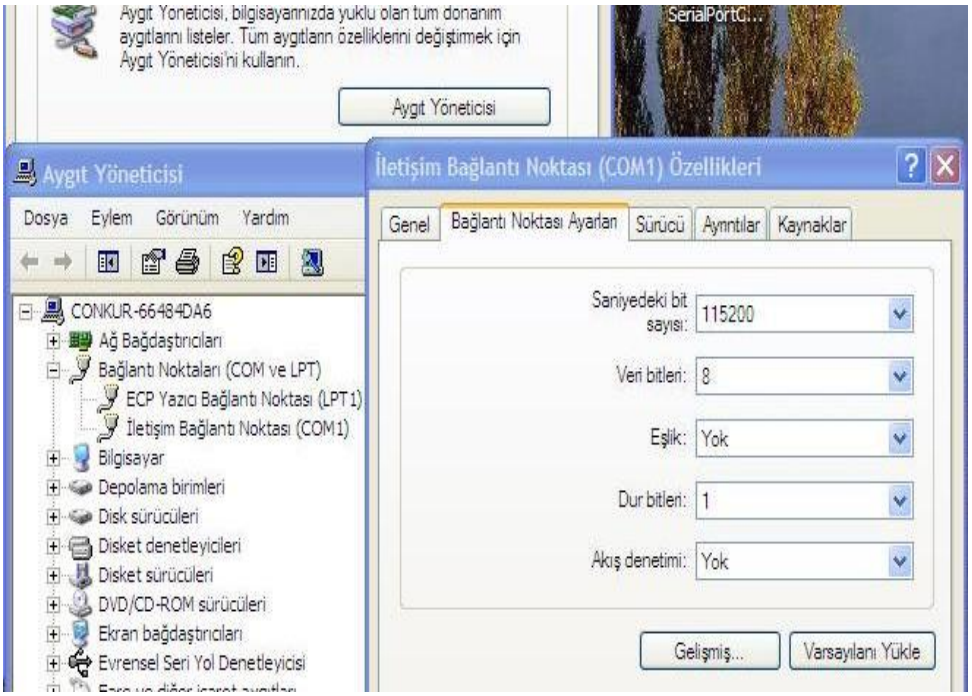
Şekil 3.21 : RS–232 bağlantı kablosu

PC ile Mini 2440 gömülü sistem arasında RS–232 haberleşmesi yapılmak istenmiştir. RS–232 bağlantı kablosu ile iki aygıt birbirine bağlanmıştır (Şekil 3.22).



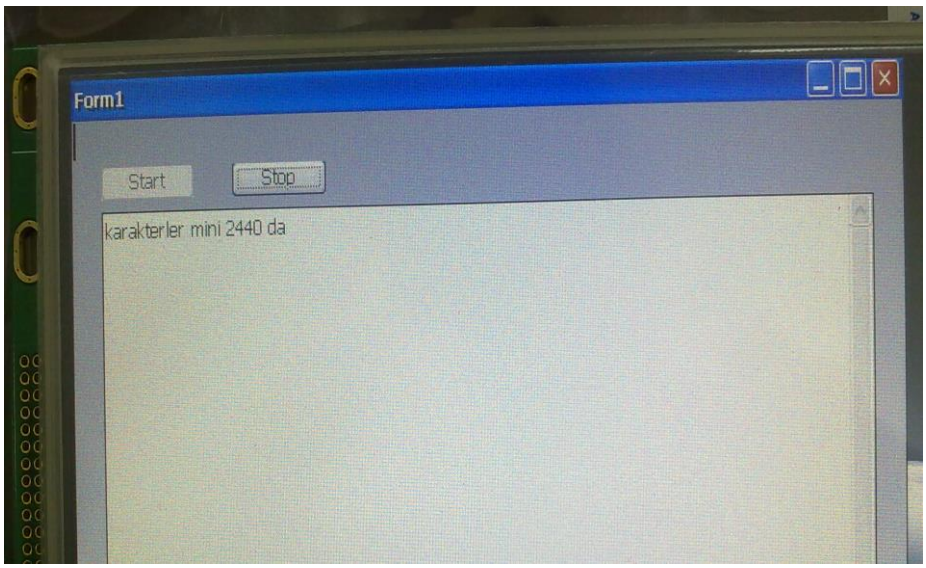
Şekil 3.22 : İki aygıtın birbirine bağlanması

Daha sonra PC'den Bilgisayarım-Özellikler-Aygıt Yöneticisi kısmına girilerek Şekil 3.23'de görüldüğü gibi COM1 portunda bağlantı sağlandığı tespit edildikten sonra Bağlantı Noktası Ayarları sekmesinden saniyedeki bit sayısı 115200 olarak ayarlanmıştır.



Şekil 3.23 : Bağlantı ayarlarının yapılması

Bağlantı ayarlarının yapılmasından sonra her iki aygıttaki yazılımda 'Start' Düğmesine tıklanarak haberleştirme işlemi başlatılmıştır. İlk işlem olarak PC klavyesinden 'karakterler mini 2440 da' yazısı tuşlanarak bu karakterler mini 2440 gömülü sisteme gönderilmiştir. Daha sonra Mini 2440 gömülü sisteminde çalışan haberleşme yazılımında Şekil 3.24'de görüldüğü gibi 'karakterler mini 2440 da' yazısının yazdığı gözlenmiştir. Böylece PC ile mini 2440 aygıtları RS–232 haberleşme protokolü kullanılarak haberleştirilmiştir.



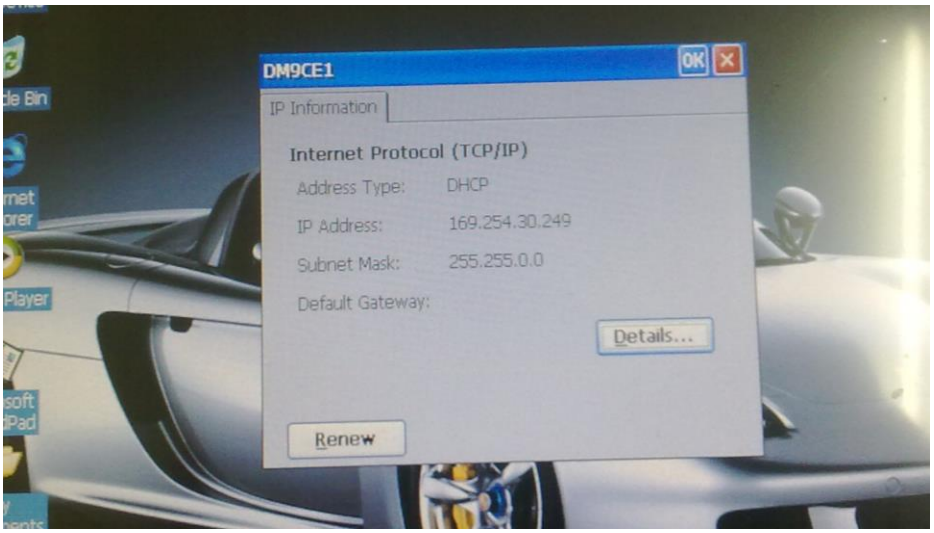
Şekil 3.24 : PC'den gönderilen karakterlerin mini 2440 tarafından algılanması

Mini 2440 gömülü sistemin klavyesinden ve harici olarak USB ile bağlanan bir klavyeden de benzer olarak ‘karakterler PC de’ yazısı tuşlanarak bu karakterler PC’ye gönderilmiştir. Daha sonra PC’de çalışan haberleşme yazılımında Şekil 3.25’de de görüldüğü gibi ‘karakterler PC de’ yazısının yazdığı gözlenmiştir. Böylece mini 2440 gömülü sistem ile de RS–232 haberleşme protokolü kullanılarak diğer bir aygıta karakterler gönderilmiştir.

Şekil 3.25 : Mini 2440 tarafından gönderilen karakterlerin PC tarafından algılanması

3.3.3.2 Ethernet haberleşme protokolü

Mini 2440 gömülü sistemde Ethernet haberleşmesi yapabilmek için gerekli donanım bulunmaktadır. Fakat daha önce kullanılan test programlarında Ethernet portunun çalışıp çalışmadığı ile ilgili bir yazılım bulunmadığından Ethernet portunun çalışıp çalışmadığı bilinmemekteydi. Yazılımı yazmadan önce bu portun çalışıp çalışmadığı test edilmiştir. Bu test için PC’de bulunan Ethernet portu ile Mini 2440 gömülü sistemin Ethernet portu arasında Ethernet kablosu yardımıyla Ethernet haberleşmesi sağlandı. Haberleşme sağlandıktan sonra Mini 2440 gömülü sistemin bir IP adresi aldığı gözlenmiştir (Şekil 3.26).

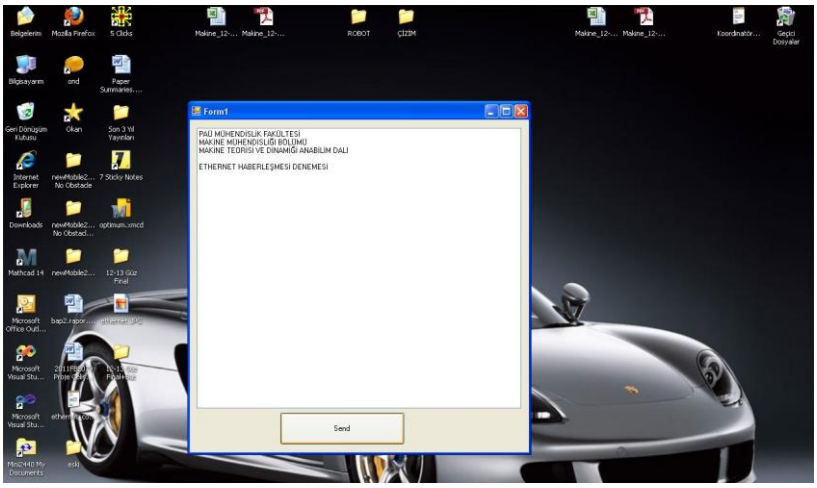


Şekil 3.26 : Mini 2440 ethernet bağlantısı

Mini 2440 gömülü sistemin Ethernet portunun çalıştığının anlaşılmasından sonra bu haberleşme protokolü kullanılarak Mini 2440 ile PC nin haberleşmesini sağlamak amacıyla Microsoft Visual Studio 2008 programı kullanılarak PC ve Mini 2440 gömülü sistem için ayrı yazılımlar geliştirilmiştir. Daha sonra Şekil 3.27de görüldüğü gibi iki aygıt birbirine ethernet kablosu ile bağlanmıştır.

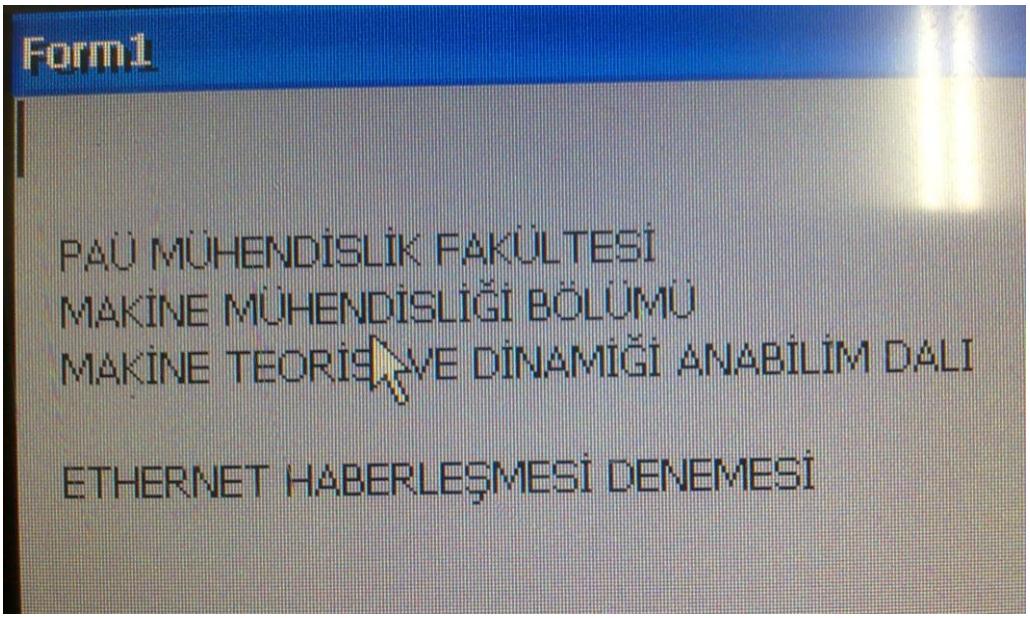
Şekil 3.27 : Aygıtların ethernet kablosu ile birbirine bağlanması

İki aygıt birbirine bağlandıktan sonra PC'den Mini 2440 gömülü bilgisayara Şekil 3.28'de görülen karakterler "Send' düğmesine tıklanarak gönderilmiştir.



Şekil 3.28 : PC'den karakterlerin gönderilmesi

Gönderilen bu karakterler mini 2440 gömülü sistemi için geliştirilen yazılımda bire bir algılanmıştır (Şekil 3.29). Ethernet haberleşme protokolu kullanılarak Mini 2440 ile PC arasında haberleşme sağlanmıştır.

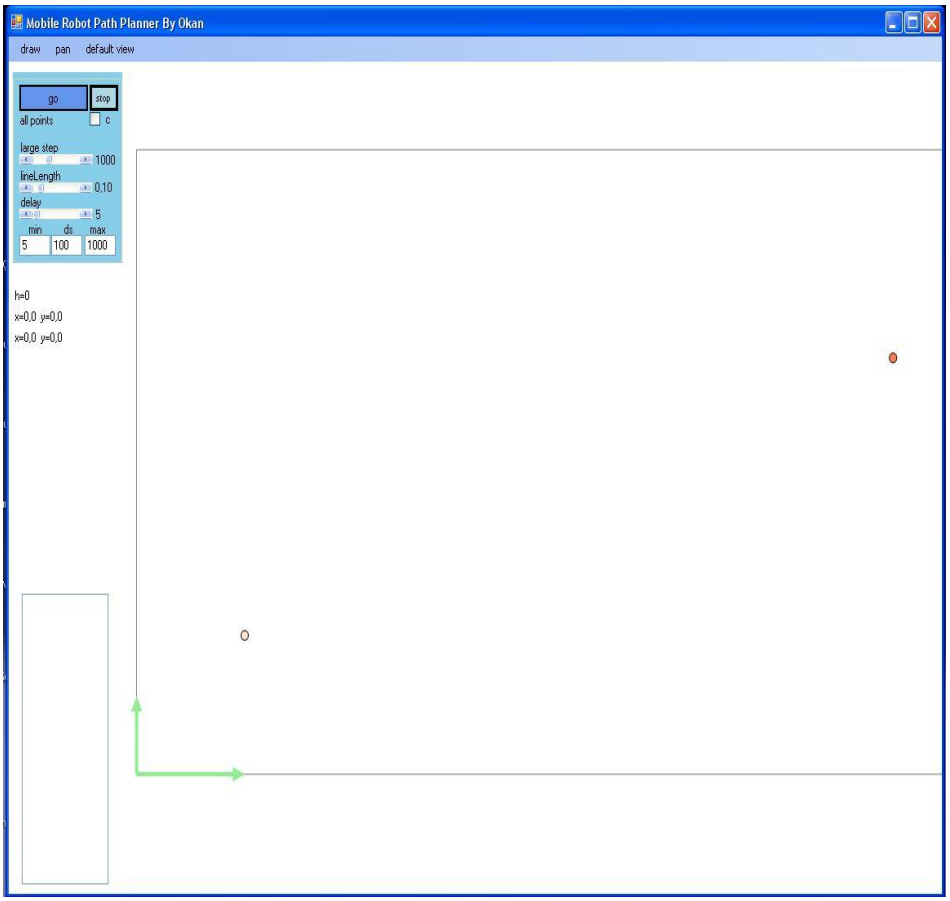


Şekil 3.29 : Mini 2440 gömülü sistemin gönderilen karakterleri algılaması

3.3.4 Mobil robot hareket yazılımının geliştirilmesi

Mini 2440 gömülü sistemde ikinci bölümde detaylıca anlatılan, geliştirilmiş olan mobil robot hareket yazılımları çalışmamaktadır. Sebebi Mini 2440 gömülü sistemin içerisinde yüklü olan Microsoft Windows CE işletim sisteminin Microsoft Compact Framework yazılımı ile kodları derleyebiliyor olmasıdır. İkinci bölümde detaylıca anlatılmış olan yazılımlar ise Microsoft Net Framework yazılımı ile kodları derlenebilecek şekilde yazılmıştır. Dolayısıyla Mini 2440 gömülü sistemde mobil

robot hareket yazılımının çalışması için kod derleyicisi olarak Microsoft Compact Framework yazılımının kullanıldığı bir yazılım yeniden geliştirilmiştir. Mini 2440 gömülü sistemin işlemci hızı ve RAM miktarı günümüzdeki bilgisayarlar ile kıyaslandığında biraz daha az miktardadır. Dolayısıyla bilgisayar için geliştirilmiş yazılımın bütün özelliklerinin kullanılması Mini 2440 gömülü sistem için geliştirilen yazılımdaki işlemlerin yavaş gerçekleştirilmesine ve hatta yazılımdaki kullanıcı arayüzünün kullanılamamasına sebep olması muhtemel bir varsayımdır. Bu sebepten ötürü öncelikle bilgisayar için geliştirilmiş yazılımda kod incelemesi yapılmıştır. Gereksiz kodlar ve işlemler iptal edildikten sonra kod optimizasyonu yapılarak yazılım hem düzenli hem de sade bir hale getirilmiştir (Şekil 3.30).

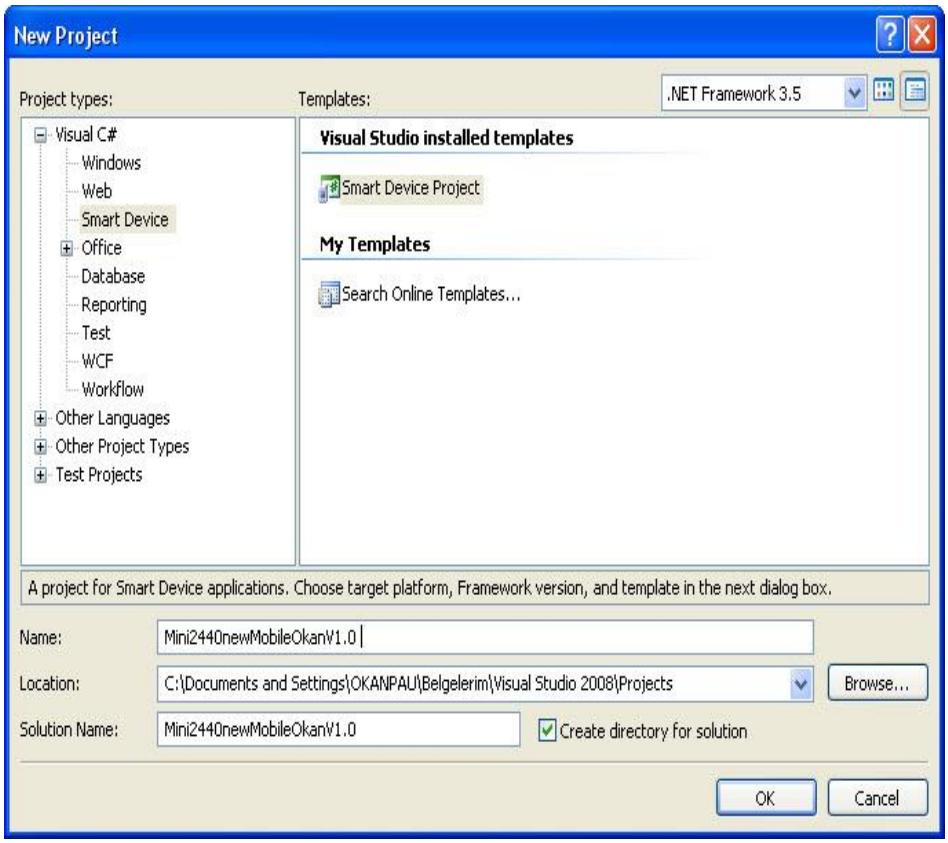


Şekil 3.30 : Mobil robot yazılımının sadeleştirilmesi

Geliştrilmiş olan bu sade yazılımın Mini 2440 gömülü sistemde çalışabilmesi için Microsoft Visual Studio 2008 programında kod derleyicisi olarak Microsoft Compact Framework yazılımının seçilmiş olduğu yeni bir proje oluşturularak yeni bir mobil robot hareket yazılımı geliştirilmiştir.

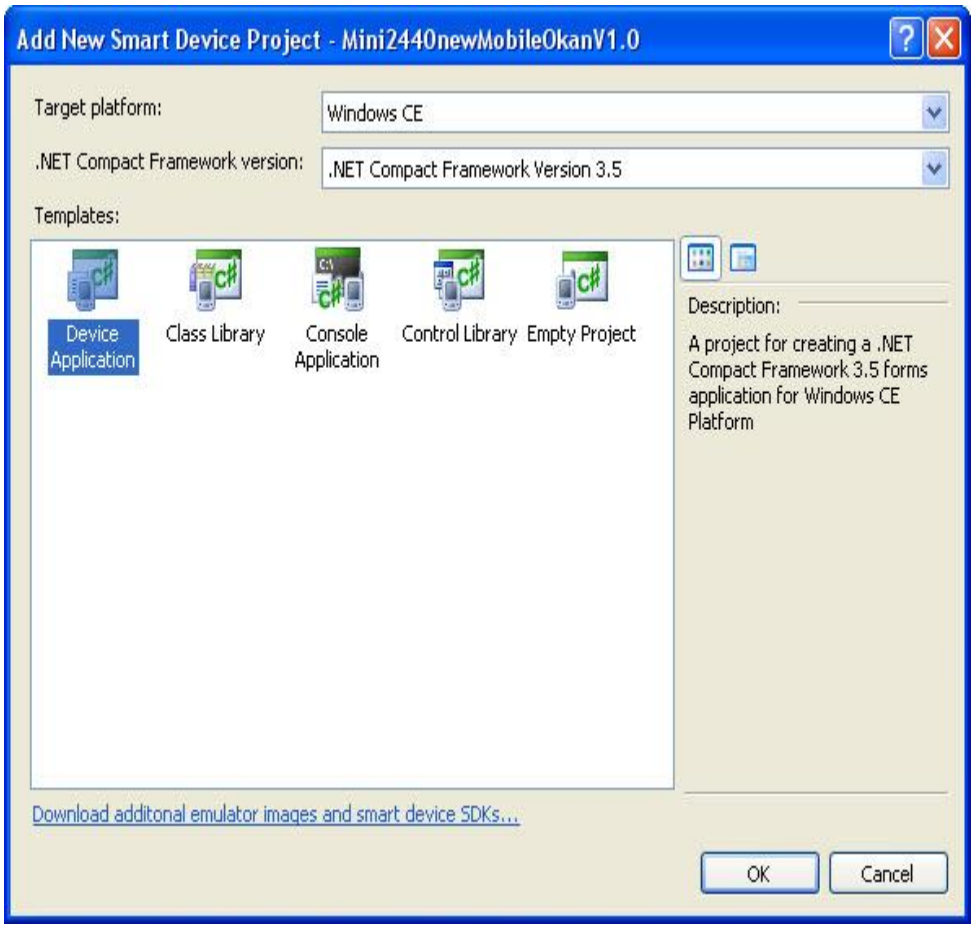
3.3.4.1 Yeni bir projenin oluşturulması

Öncelikle Microsoft Visual Studio 2008 programı çalıştırılarak File-New-Project sekmesine tıklanmıştır. Programlama dili olarak Visual C# seçildikten sonra Smart Device sekmesinde bulunan Smart Device Project seçilmiştir (Şekil 3.31).



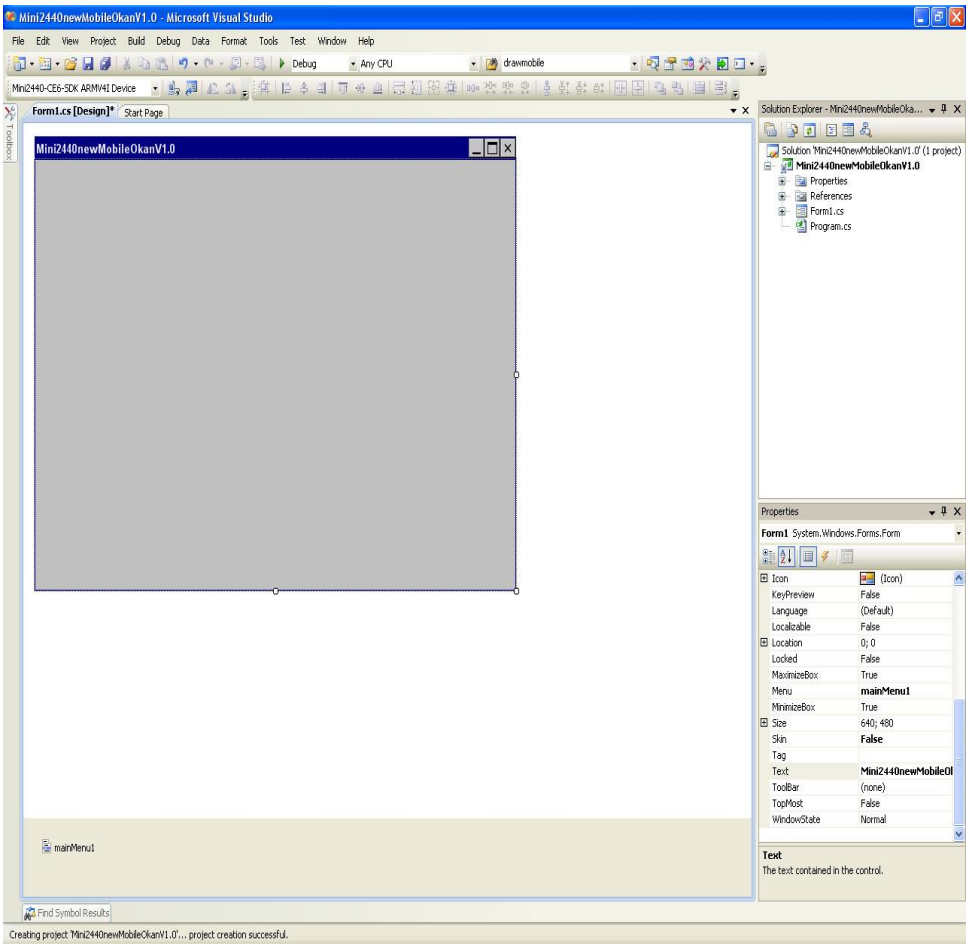
Şekil 3.31 : Yeni bir projenin oluşturulması

Projenin ismi girildikten sonra OK butonuna tıklanmıştır. Daha sonra ekranda beliren pencerede Şekil 3.32'de görüldüğü gibi Target Platform olarak Windows CE ve kod derleyicisi olarak da Microsoft Compact Framework V3.5 seçilmiştir.



Şekil 3.32 : Akıllı cihaz ayarlarının girilmesi

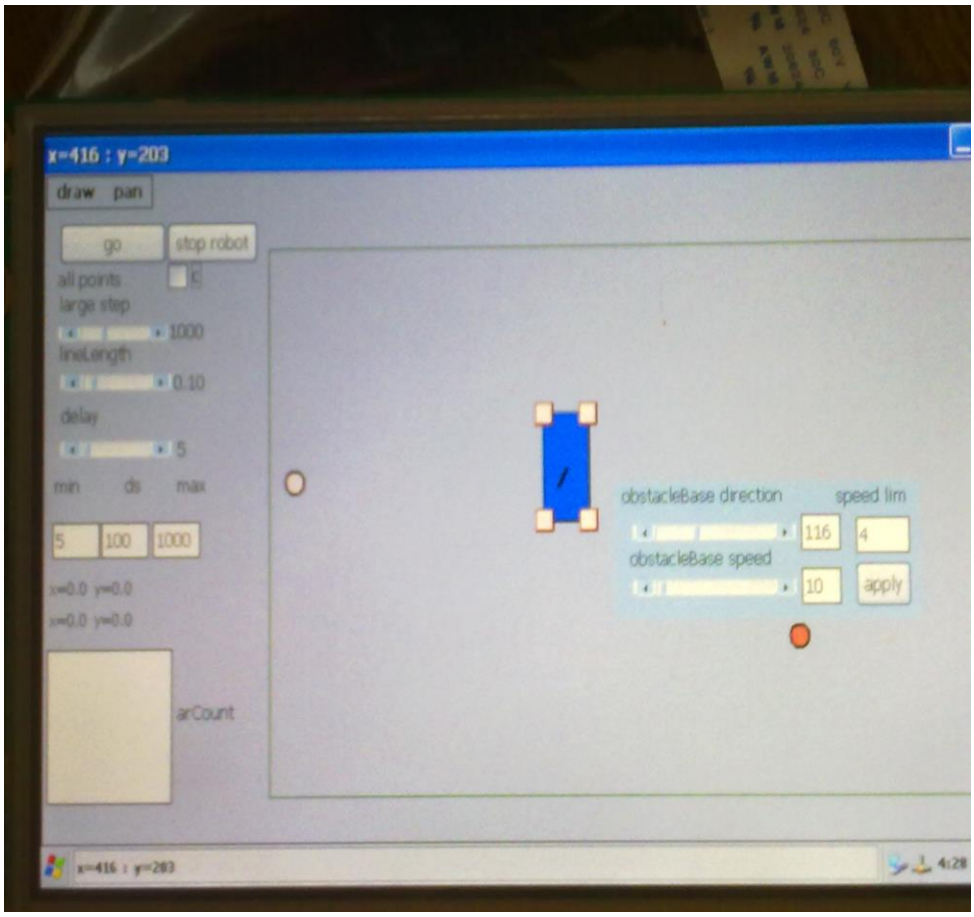
En son işlem olarak Device Application seçilmiştir ve OK butonuna tıklanarak yeni bir proje oluşturulmuştur (Şekil 3.33).



Şekil 3.33 : Oluşturulmuş olan yeni proje

3.3.4.2 Mini2440newMobileOkanV1.0 yazılımı

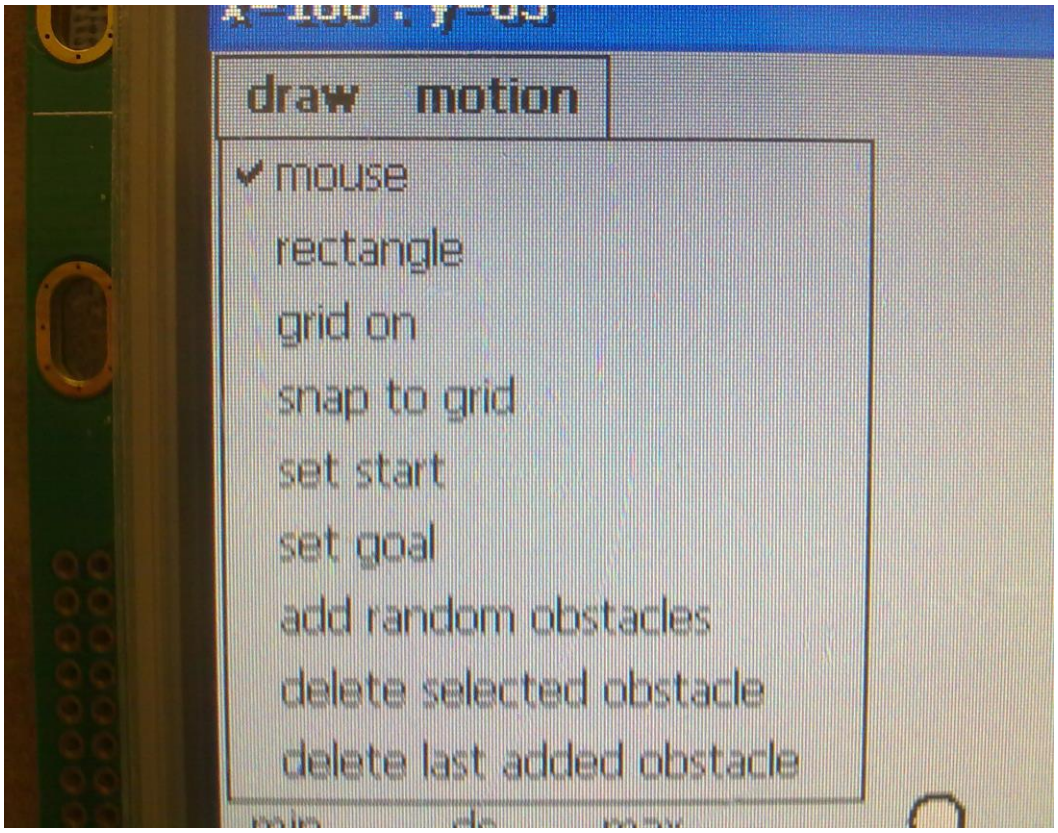
Oluşturulan projeye sade hale getirilen yazılımın kodları aynen eklenmiştir. Daha sonra kod derleyicilerin farklı olmasından kaynaklanan 1000 e yakın kod tanınmama hatası teker teker incelenerek çözülmüştür. Hatalı kodlar Compact Framework kod derleyicisinin derleyebildiği, algılayabildiği kodlarla değiştirilerek, değiştirilemeyen kodların da silinmesi ile bu sorunlar detaylı bir çalışmanın neticesinde çözülmüştür. Yeni oluşturulmuş olan projede F5 tuşuna basılarak geliştirilen yazılım Şekil 3.34'de görüldüğü gibi Mini 2440 gömülü sistemde sorunsuz çalıştırılmıştır.



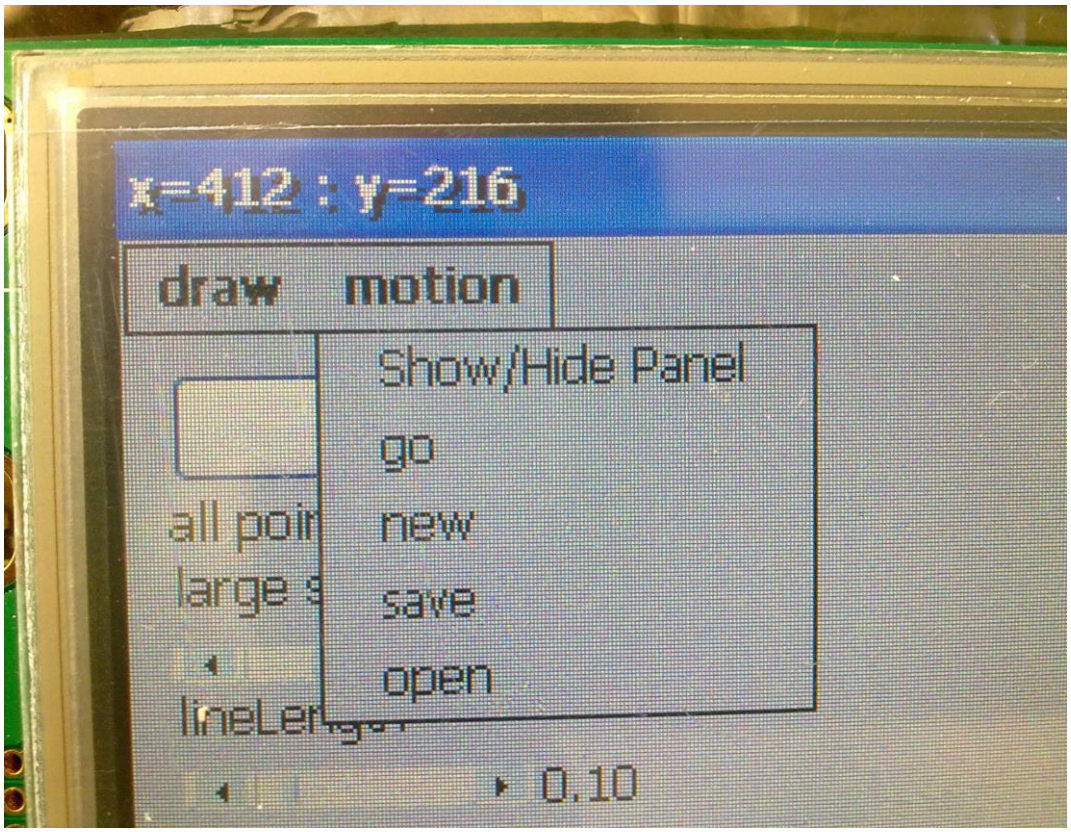
Şekil 3.34 : Mini 2440 gömülü sistem için geliştirilen yazılım

3.3.4.3 Mini2440newMobileOkanV1.1 yazılımı

Bu yazılımda kod derleyicisi uyumsuzluğundan dolayı önceki yazılım geliştirilirken silinen kodların göreceği işlevi görebilecek alternatif algoritmalar geliştirilmeye çalışılmıştır. Mobil robotun hareketine başladığı yer ile ulaşacağı hedefin yerlerinin değiştirilmesinin sağlandığı yeni menüler ve en önemlisi yapılan çizimleri yani hareket alanını kaydetme özelliği yeni bir algoritma geliştirilerek eklenmiştir. Yazılıma draw ve motion isimli iki adet menü eklenmiştir. Şekil 3.35'de görüldüğü gibi draw isimli menüde çizim ile ilgili olan işlevler, motion isimli menüde de mobil robotun hareketi ve hareket ortamı ile ilgili işlevler bulunmaktadır (Şekil 3.36). Bu menülerdeki komutların işlevleri Tablo 3.1 ve Tablo 3.2'de kısaca açıklanmıştır.



Şekil 3.35 : Draw menüsü



Şekil 3.36 : Motion menüsü

Tablo 3.1 : Draw menüsü komutlarının işlevleri

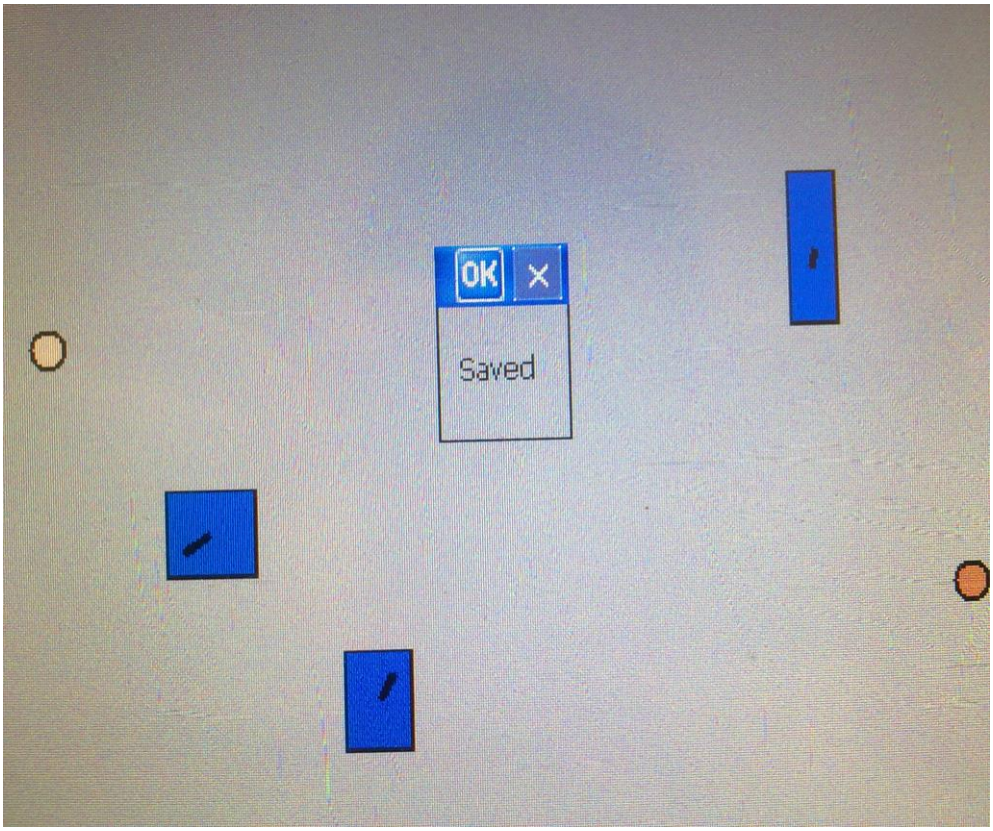
Komut	İşlevi
mouse	Ekranda çizili olan engelleri seçmektir.
rectangle	Ekrana dikdörtgen engel çizmektir.
grid on	Grid çizgilerini ekrana çizmektir.
snap to grid	Grid çizgilerini algılamaktır.
set start	Başlangıç noktasını değiştirmektir.
set goal	Hedef noktasını değiştirmektir.
add random obstacles	Ekrana rastgele engeller eklemektir.
delete selected obstacle	Seçilen engeli silmektir.
delete last added obstacle	Son çizilen engeli silmektir.

Motion isimli menüde ise mobil robotun hareketi ile ilgili işlevler bulunmaktdır (Şekil 3.43).

Tablo 3.2 : Motion menüsü komutlarının işlevleri

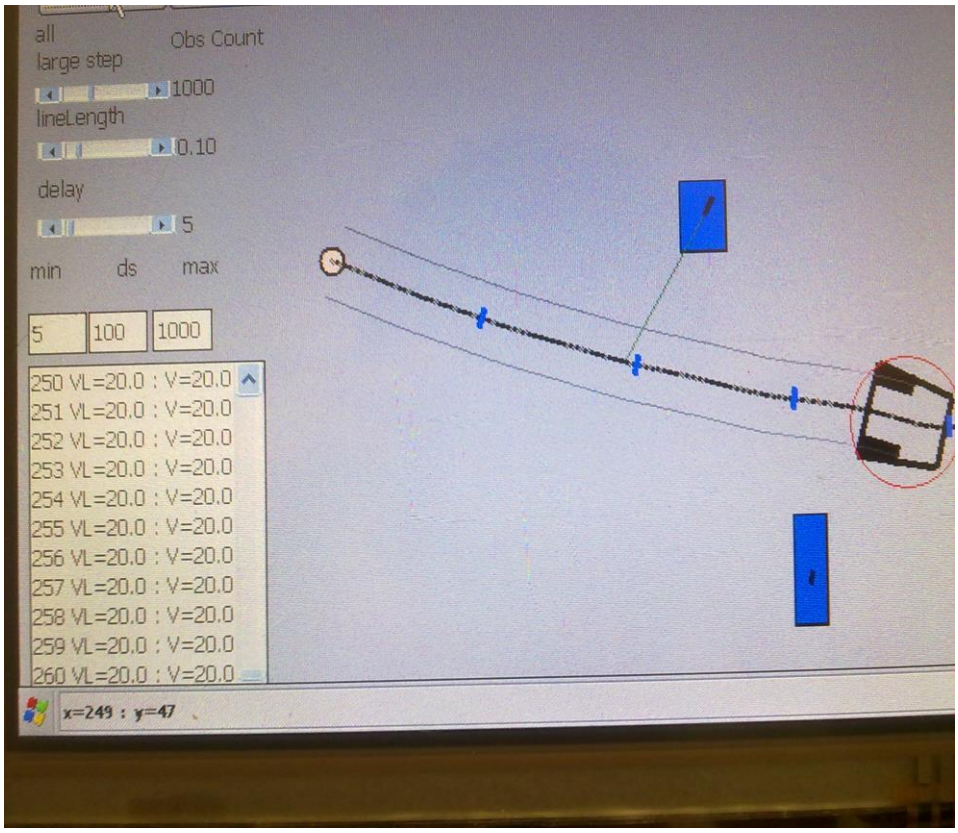
Komut	İşlevi
Show/Hide Panel	Ayarlar panelini gösterip gizlemektir.
go	Hareketi başlatmaktır.
new	Boş bir hareket alanı oluşturmaktır.
save	Hareket alanını kaydetmektir.
open	Kayıtlı olan hareket alan dosyasını açmaktır.

Hareket alanının kaydedilmesi çizilen engelin boyutunun, hareket doğrultusunun ve hızının, mobil robotun harekete başladığı konumunu ve ulaşması gereken hedefin konumunu, hız sınırlarını ve grid çizgilerinin gösterilip gösterilmediği gibi ayarları dolayısıyla yazılımda yapılmış olan bütün değişiklikleri kaydetmektedir. Kaydedildikten sonra ekranda kullanıcıya bildirmek amacıyla bir mesaj kutusu gösterilmektedir (Şekil 3.37).



Şekil 3.37 : Hareket alanının kaydedilmesi

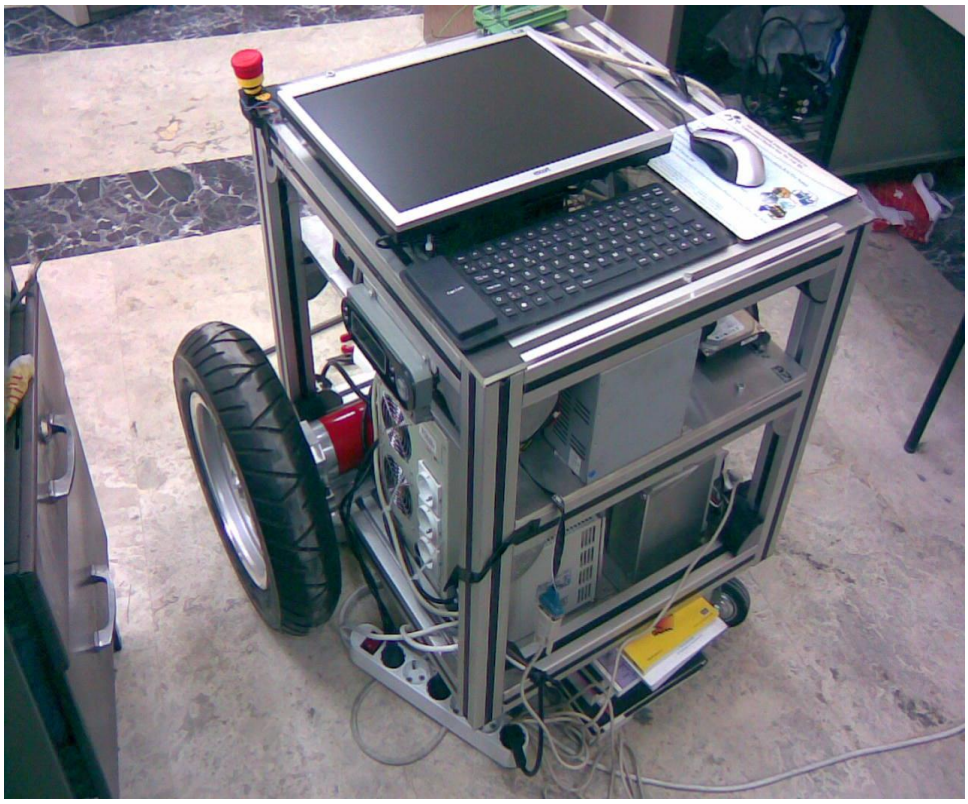
Bu yazılımda mobil robotun hedefine varması için takip etmesi gereken yol potansiyel alan yöntemi kullanılarak planlanmıştır. Planlanmış olan bu yolu takip edebilmesi için mobil robotun sol ve sağ tekerlerinin hızları hesaplanarak sol alttaki liste kutusuna eklenmiştir (Şekil 3.38). Bu değerler RS-232 haberleşmesi ile STM32F4 Discovery Board cihazına gönderilerek tekerlere bağlı olan motorlara bu hızlarda dönme emri gönderilmiştir ve mobil robotun planlanmış olan yolu takip ederek hedefine ulaştığı görülmüştür.



Şekil 3.38 : Tekerlerin hızlarının liste kutusunda belirtilmesi

4. MOBİL ROBOTUN TASARIMI

Bu tezde daha önce imalatı gerçekleştirilmiş olan mobil robotun [13] hareket kabiliyetinin arttırılması amaçlanmıştır (Şekil 4.1). Bu amaçla mobil robotun tasarımında iyileştirmeler yapılarak mobil robotun hareket kabiliyeti arttırılmaya çalışılmıştır. Tasarımda yapılan yeniliklerde hafif parçalar kullanılmaya çalışılarak mobil robotun ağırlığının azaltılması ve motorları kontrol etmekte kullanılan hareket kontrol ünitesinde yenilikler yapılarak mobil robotun istenilen hızlarda hedefine varması sağlanmıştır.



Şekil 4.1 : Daha önce imalatı gerçekleştirilmiş olan mobil robot

4.1 Mobil Robotun Tasarımında Yapılan İyileştirmeler

Mobil robotun tasarımında toplam ağırlığı azaltmak ve hareket kabiliyetini arttırmak amacıyla iyileştirmeler yapılmıştır

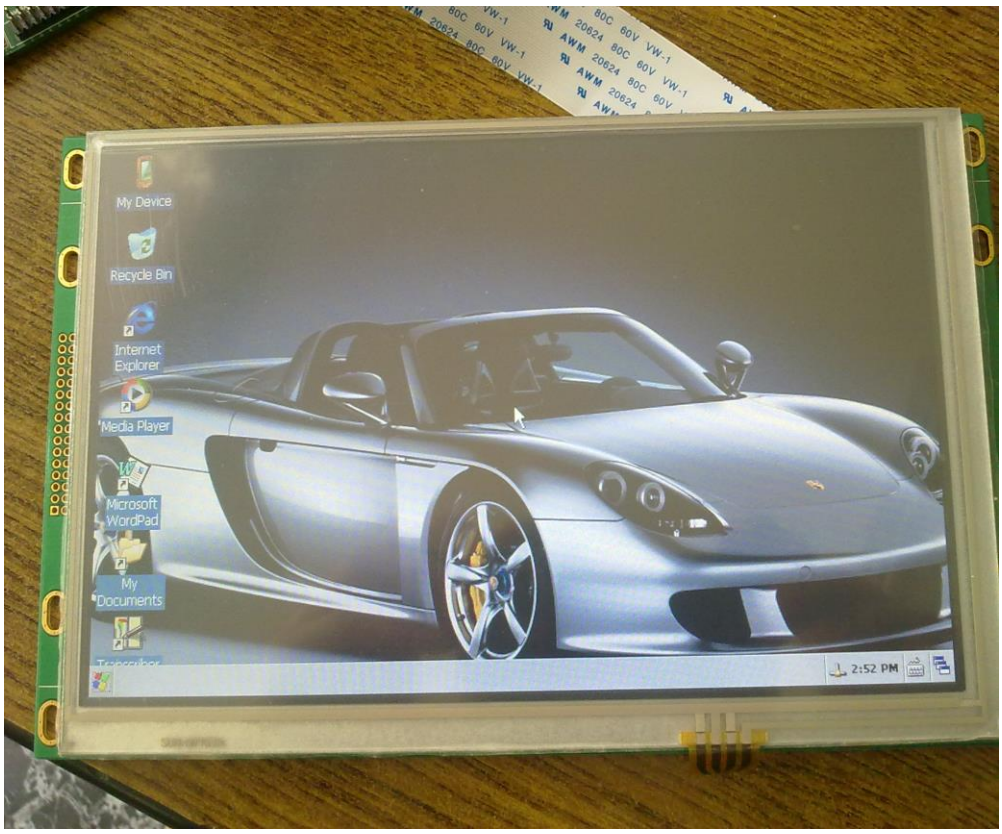
4.1.1 Toplam ağırlığın azaltılması

Mobil robotun tasarımında kullanılan temel elemanlar aşağıdaki parçalardan oluşmakta idi.

- 1 adet bilgisayar
- Profil kafes
- UPS
- 2 adet servo sürücü
- 2 adet servo motor
- 2 adet redüktör
- 1 adet kablolama kartı
- 1 adet 24V güç kaynağı
- 2 adet motor tekeri
- 2 adet avare teker
- 1 adet monitör

Mobil robotun tasarımında ağırlığı azaltmak amacıyla bilgisayar ve monitör yerine üçüncü bölümde detaylıca anlatılmış olan Mini2440 gömülü sistem kullanılmıştır. Mini2440 gömülü sistem bilgisayarın yapabildiği işlemleri gerçekleştirebilen 32 bit ARM işlemcisine ve usb, ethernet, SD hafıza kartı vb. girişlere sahip olan bir cihazdır. Bu cihaz için üçüncü bölümde detaylıca anlatıldığı üzere Microsoft Visual Studio NET programı kullanılarak mobil robotun engellerden kaçınarak hedefine ulaşabilmesi için gerekli olan hızların hesaplandığı ve hareketin simüle edildiği yazılımlar geliştirilmiştir. Bu yazılımlar kullanılarak mobil robotun engellere çarpmadan hedefine ulaşması sağlanmıştır. Bu yapılan iyileştirmeler ile mobil robottaki bilgisayar anakartının, anakarta bağlı olan cihazların, mobil robotun üzerine eklenen monitorun ağırlığından kurtulunmuştur ve mobil robotun dış görünüşündeki

karmaşıklık azaltılmıştır. Monitor olarak ta Mini 2440 gömülü sistmin 7 inçlik ekranı kullanılmıştır (Şekil 4.2).



Şekil 4.2 : 7 inçlik ekran

Mobil robotun üzerinde bulunan cihazlara, gerilimi 220V olan elektriği sağlayan UPS güç kaynağı yerine binek araçlarda kullanılan gerilimi 12V akımı 72 Amper olan bir tane akü kullanılmıştır (Şekil 4.3). Akü kullanılmasının sebebi UPS şarjının çok kısa sürede bitmesi akünün ise şarjının daha uzun süre dayanabiliyor olmasıdır. Ayrıca akü mobil robotta UPS'e nazaran daha az yer kaplamaktadır ve daha hafiftir.

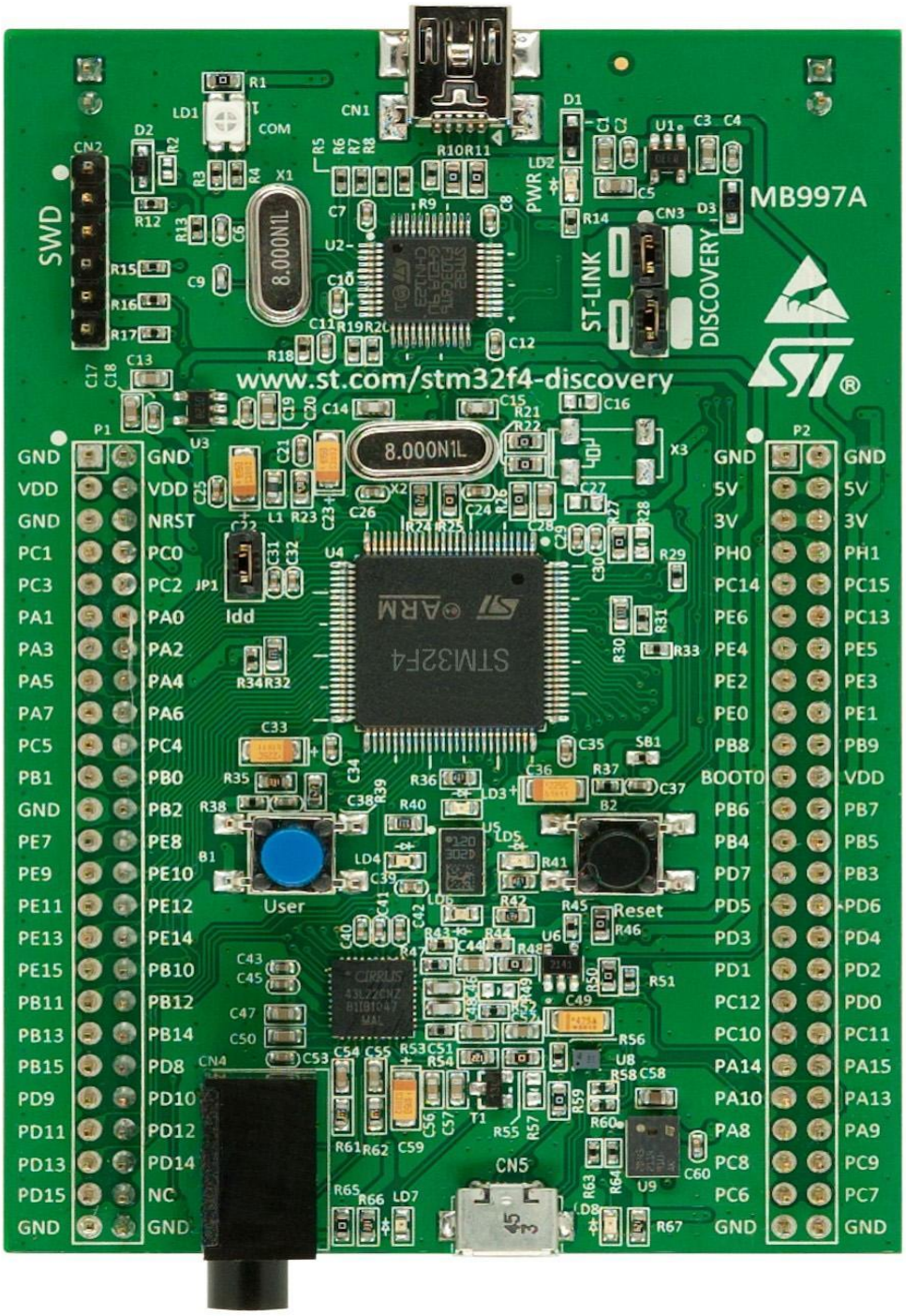
Şekil 4.3 : 12V akü

Akünün 12V olan gerilimi 220V gerilim değerine bir dönüştürücü ara parça sayesinde dönüştürülerek mobil robottaki 220V gerilime ihtiyaç duyan cihazlar çalıştırılmıştır (Şekil 4.4).

Şekil 4.4 : 12V-220V dönüştürücü

Mobil robot hareket kontrol ünitesinde, bilgisayarın PCI girişine takılmak suretiyle kullanılan MPC3034 hareket kontrol kartının ağırlığından kurtulunmuştur. Bu kart yerine STM32F4 Discovery Board kullanılmıştır. ST firmasının satışa sunduğu Şekil 4.5'de görüldüğü gibi STM32F4 Discovery Board cihazının üzerinde ARM Cortex

M4 tabanlı 168 MHz'lik bir mikrodenetleyici bulunmaktadır. M4 çekirdeğinde ise M3′e ekstra olarak bir de FPU (Floating Point Unit) bulunmaktadır. Çip içerisinde FPU'nun bulunması ondalıklı işlemler için tasarlanmış ekstra bir birim anlamına gelmektedir.



Şekil 4.5 : STM32F4 Discovery Board

4.1.2 Toplam maliyetin azaltılması

Toplam ağırlığın azaltılması amacıyla mobil robotun tasarımında yapılan iyileştirmeler aynı zamanda toplam maliyetinde azalmasına sebep olmuştur. Bilgisayar anakartı ve monitor yerine kullanılan Mini 2440 gömülü sistem ve 7 inçlik ekran, UPS yerine kullanılan binek araçlarda bulunan 12V luk akü, 12V-220V dönüştürücüsü daha ucuz bir fiyata kolaylıkla temin edilebilmektedir. Ayrıca bu cihazlar eski tasarımda kullanılanlara göre daha az güç harcamaktadırlar. Bu iyileştirmeler sayesinde mobil robotun imalatının maliyeti önemli ölçüde azaltılmıştır. Mobil robotun hassasiyeti arttırılarak kompakt bir görünüme sahip olması sağlanmıştır.

4.1.3 Hareket kabiliyetinin arttırılması

Daha önce imal edilmiş olan mobil robotta MPC3034 hareket kontrol kartı kullanılarak motorlar döndürülerek mobil robotun hedefine ulaşması sağlanmaktaydı. Mobil robotun hareketi kontrol kartının kapasitesiyle sınırlı olmaktaydı. Kart üretici firmasının yazıdığı kod parçacıkları kullanılarak motorlara hareket emri gönderilebilmekteydi. Mobil robotta pozisyon kontrolü yapılamamıştı ve bu durum mobil robotun hareket ederken zıplamasına neden olmaktaydı. Bu sorun hız kontrolü yapılmak suretiyle çözülmüştü. Yeni tasarımda hareket kabiliyetini arttırmak amacıyla hareket kontrol ünitesinin gerçekleştirdiği işlemleri gerçekleştirebilecek işlemci kapasitesine sahip olan STM32F4 Discovery Board kullanılmıştır. Bu board üzerinde bulunan mikrodenetleyici programlanarak motorlara emir gönderilmesi sağlanmıştır. Mini 2440 gömülü sistemde çalıştırılan yazılımda hesaplanan ve mobil robotun, engellerden kaçınarak hedefine varması için yapılan yol planlamasına bağlı olarak hareket etmesi için gerekli olan mobil robotun sol ve sağ tekerlerinin dönme hızları RS-232 haberleşmesi yardımıyla STM32F4 Discovery Board cihazına gönderilmiştir. Daha sonra bu değerlere bağlı olarak motorlara emir gönderilmesi sağlanarak mobil robotun planlanan yol üzerinde hareketi temin edilmiştir. Hareket kontrol ünitesinde yapılan bu değişiklikten sonra mobil robotun pozisyon kontrolünde ortaya çıkan zıplama problemi de ortadan kaldırılmıştır.

4.2 Mobil Robotun Donanımı

Mobil robotun donanımı temel olarak aşağıdaki parçalardan oluşmaktadır.

- 1 adet Mini 2440 gömülü sistem ve 7 inç ekran
- Profil kafes
- 12V 72A akü
- 2 adet servo sürücü
- 2 adet servo motor
- 2 adet redüktör
- 1 adet STM32F4 Discovery Board
- 1 adet 3'lü priz
- 2 adet motor tekeri
- 1 adet avare teker

Yukarıdaki parçalara ek olarak profil kafeste alt, orta ve üst plaka bulunmaktadır. Ayrıca plakalara bazı parçaların tutturulabilmesi amacıyla ek parçalar kullanılmıştır. Profil kafes Şekil 4.6'de gözükmektedir.

Şekil 4.6 : Profil kafes sistemi

Daha önce imal edilen mobil robotta [13] kullanılmış olan Delta Electronics, Inc. firmasının ASD-B0421-A model servo sürücüsü (Şekil 4.7) ile ECMA-C30604GS model servo motoru (Şekil 4.8) bu tez çalışmasında da kullanılmıştır.

Şekil 4.7 : Servo sürücü

Şekil 4.8 : Servo motor

Servo sürücü ve servo motor 400 W'lıktır. Servo motora güç servo sürücü ile sağlanmıştır. Ayrıca servo sürücü, servo motora kontrol sinyallerinin yollanmasını sağlamaktadır. Servo motorun en, derinlik ve yükseklik değerleri 60x60x160mm şeklindedir. Servo sürücünün ise en, derinlik ve yükseklik değerleri 59x147x162mm şeklindedir. Servo motor 1,6 kg ve servo sürücü 1,2 kg ağırlığındadır. Servo sürücüye hem STM32F4 Discovery Board ile hem de servo motor ile kablo bağlantısı yapılmıştır. Servo motora ise sadece servo sürücü ile kablo bağlantısı yapılmıştır. Servo motor 360°'yi 10000 adıma bölebilmektedir. Böylece servo motorun bir adımı 0.036°'ye karşılık gelmektedir ve bu durum servo motorun hassasiyetinin yüksek olduğunu göstermektedir.. Servo motorun torku 1,27 Nm'dir. Fakat bu tork değeri ayrıca redüktör ile güçlendirilmiştir. Böylece mobil robot için yeterince güçlü bir teker döndürme sistemi elde edilmiştir.

Redüktör olarak daha önce imal edilen mobil robotta [13] kullanılmış olan planetRolL firmasının PD085 – eAH025 – 1AA0 model redüktörü (Şekil 4.9) kullanılmıştır. Redüktör servo motorun torkunu 25 kat daha güçlendirmektedir. Böylece servo motorun 1,27 Nm olan tork değeri 1,27*25=31,75 Nm olmaktadır. 2 motor olduğu için toplam tork 2*31,75=63,5 Nm olmaktadır. Redüktörün ağırlığı 3,5kg'dır. Redüktörün boşluk miktarı ise 15 dakikadır.

Şekil 4.9 : Redüktör

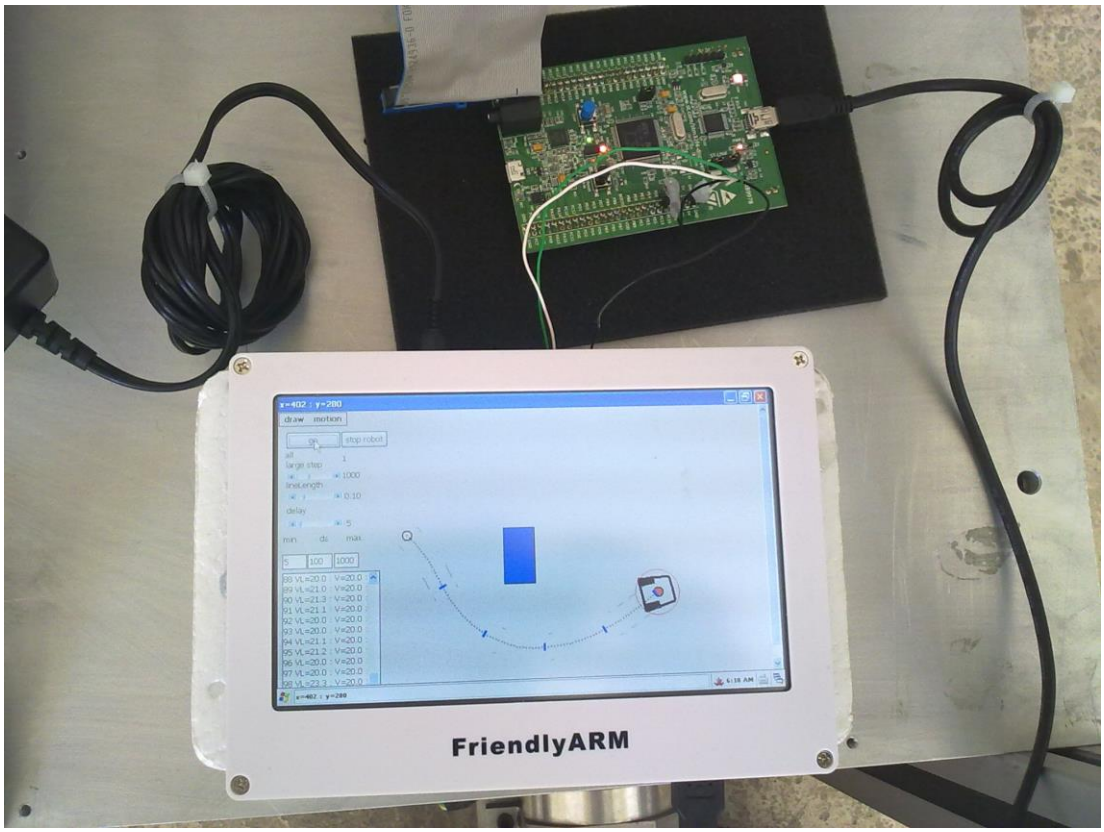
Mobil robotun üzerinde bulunan cihazlara elektrik 12V 72 amperlik binek araçlarda kullanılan akünün kutup başlarından sağlanan 12V luk gerilimin 12V-220V dönüştürücüsü ile 220 V değerine dönüştürülmesi ile sağlanmaktadır (Şekil 4.10).

Şekil 4.10 : Aküden 12V elde edilmesi

Dönüştürücüde bulunan prize bir üçlü priz takılmak suretiyle Mini 2440 ve iki adet servo sürücüye elektrik sağlanmıştır (Şekil 4.11).

Şekil 4.11 : Cihazlara elektrik sağlanması

STM32F4 Discovery Board cihazına ise elektrik Mini 2440 gömülü sistemin usb portundan sağlanmaktadır. Mobil robotun üzerinde bulunan Mini 2440 gömülü sistem için geliştirilen yazılımdan alınan sol ve sağ teker hız bilgileri RS-232 haberleşmesi yardımıyla STM32F4 Discovery Board cihazına gönderilmektedir. Bu haberleşme RS-232 ara kablosu ile değil Mini 2440 gömülü sistemde bulunan pinlere bağlanan kablolar ile yapılmaktadır. (Şekil 4.12). Çünkü STM32F4 Discovery Board cihazında seri port çıkışı soket olarak değil pin olarak bulunmaktadır.



Şekil 4.12 : Mini 2440 ile STM32F4 haberleşmesi

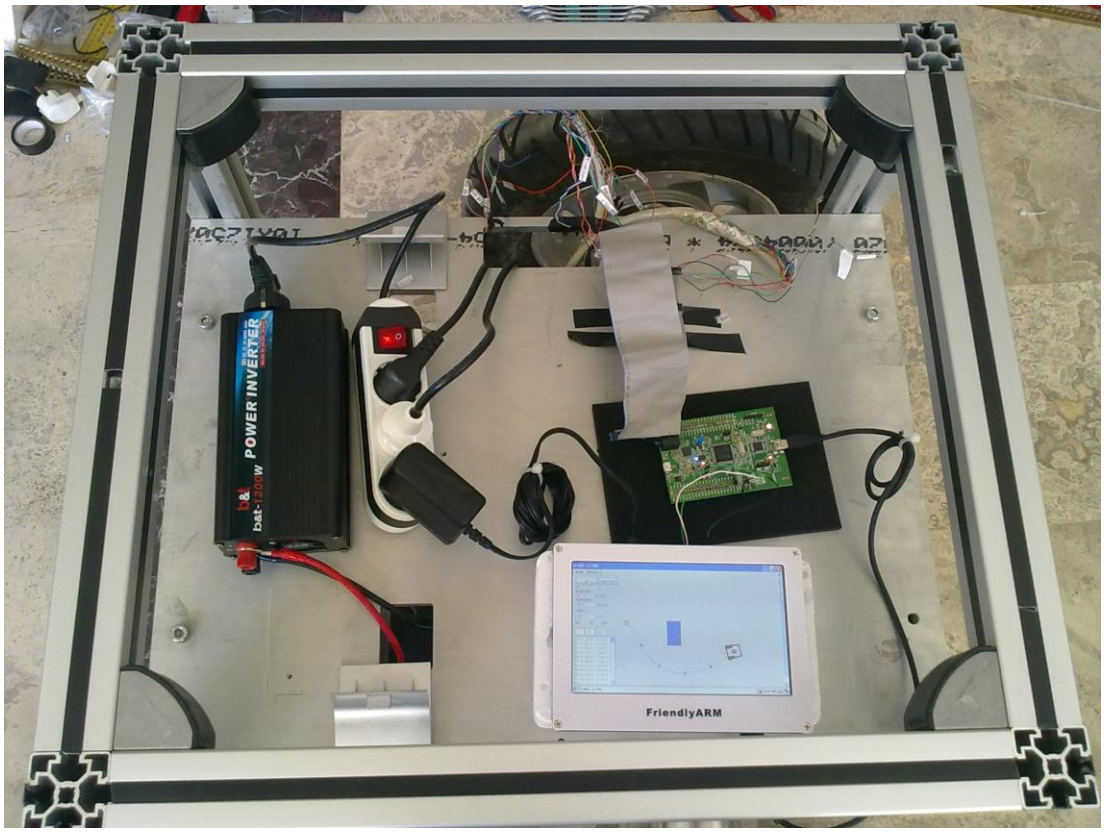
Daha sonra bu bilgiler Şekil 4.13'de görüldüğü gibi gri renkli bir ara soket yardımıyla servo sürücüye gönderilmekte oradan da servo motora iletilerek motorların istenilen hızda dönmesi sağlanarak mobil robotun hareketi sağlanmaktadır.

Şekil 4.13 : STM32F4 ile servo sürücü haberleşmesi

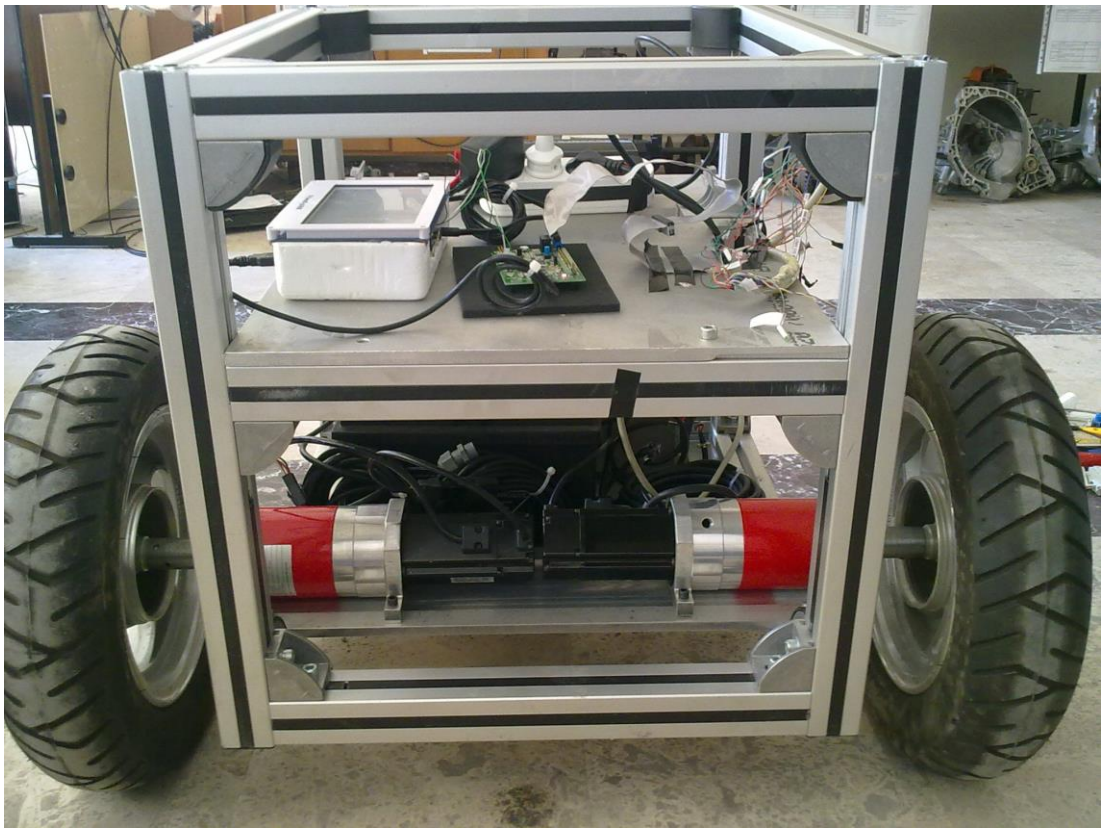
Servo motorların farklı hızlarda döndürülebiliyor olması mobil robotun manevra yapmasına olanak ve engellerden kaçınarak hedefine ulaşmasını sağlamaktadır. Bu tez çalışması sonunda montajı yapılan mobil robotun perspektif görünüşü Şekil 4.14, ön görünüşü Şekil 4.15, üst görünüşü Şekil 4.16, arka görünüşü Şekil 4.17 ve sağ görünüşü de Şekil 4.18' de gösterilmiştir.

Şekil 4.14 : Mobil robot perspektif görünüşü

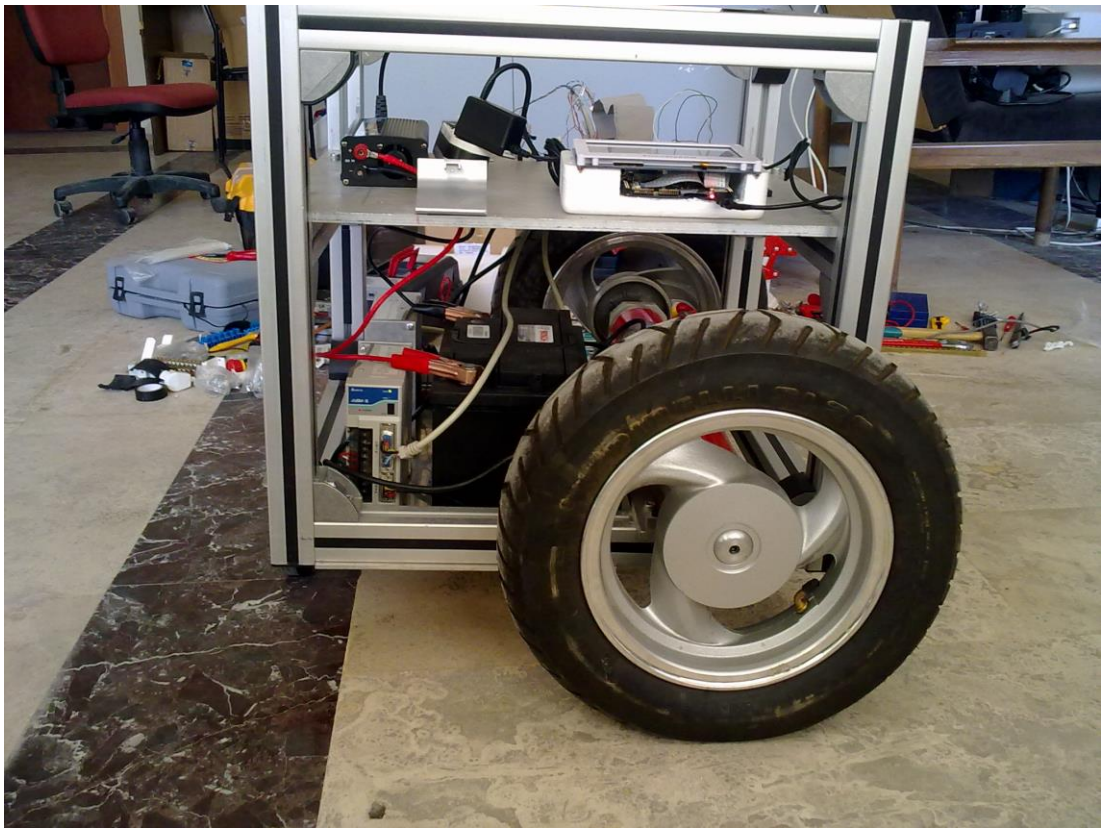
Şekil 4.15 : Mobil robot ön görünüşü



Şekil 4.16 : Mobil robot üst görünüşü



Şekil 4.17 : Mobil robot arka görünüşü



Şekil 4.18 : Mobil robot sağ görünüşü

5. SONUÇ VE ÖNERİLER

Bu tezde daha önce [13]'de tasarlanan mobil robotun tasarımında iyileştirmeler yapılarak mobil robotun hareket kabiliyeti arttırılmıştır ve hareket algoritmaları geliştirilmiştir. Hareket algoritmaları geliştirilirken otomatik yol bulma algoritması olarak global alanda çalıştığı ve local miniması olmadığı için potansiyel alan metodundan yararlanılmıştır. Sadece sabit engellerin olduğu ortamlarda potansiyel alan metodu mobil robotun yol planlamasının yapılabilmesi için yeterli olmaktadır. Fakat hareketli bir tane bile engelin bulunduğu ortamda potansiyel alan metodunun kullanılmasının yol planlaması için yeterli olmadığı gözlenmiştir. Bu çalışmada ortamdaki hareketli engelin veya engellerin konumlarının tespiti dikkate alınarak yol planlamasının yapıldığı bir yazılım Microsoft Visual Studio NET kullanılarak geliştirilmiştir. Yazılımda ekrana fare yardımıyla engeller çizilmekte bu engellere hareket doğrultuları ve hızları verilebilmektedir. Hareketli engellere çarpmadan mobil robotun hedefine ulaşması sağlanmaktadır. Çizilen engeller nokta olarak düşünülmüş, yükseklik ve genişlik değerleri ihmal edilmiştir. Yazılıma sürekli ekleme yapılarak yeni özellikler kazandırılmıştır. Yazılıma, ekranda konumu, hızı ve doğrultusu yazılım tarafından rastgele belirlenen engelin dikkate alınması ve mobil robota çarpma riski taşımayan engellerin tespit edilip dikkate alınmaması gibi yeni özellikler kazandırılmıştır. Yazılımda yapılan denemelerde mobil robotun ani hız değişimlerine maruz kaldığı gözlemlenmiştir. Daha sonra ani hız değişimlerine maruz kalmasını önlemek amacıyla mobil robotun ivmeli hareketi üzerine detaylı çalışmalar yapılmıştır.

Mobil robotun yeni tasarımında Mini 2440 gömülü sistem tercih edilerek mobil robotun hareket kontrol sistemi basitleştirilmiş ve aynı zamanda mobil robotun güç tüketimi de azaltılmıştır. Güç kaynağı olarak gerek şarj edildikten sonra uzun süre kullanılabilmesinden gerekse UPS'e nazaran daha hafif olmasından dolayı, binek araçlarda kullanılan 12V'luk 72 amperlik bir akü kullanılmıştır. Gerilim değeri 12V olan akünün gerilim değeri 12V-220V dönüştürücü yardımıyla 220V değerine

yükseltilerek, 220V gerilime ihtiyaç duyan mobil robotun üzerinde bulunan cihazlara gerekli gerilim sağlanmıştır. Mobil robotun hareket kontrol ünitesinde maliyeti çok düşük olan STM32F4 Discovery Board kullanılmıştır. Böylece hareket kontrol sistemi daha basit hale getirilmiştir. Ayrıca mobil robotun hareketini sağlayan iki adet servo motora hareket emri gönderilirken, yalnızca hareket kontrol kartı üreticisinin sağladığı kod parçacıklarının kullanılabilmesinden kaynaklanan hareket kabiliyetinin kartın kapasitesi ile sınırlı olması durumu ortadan kaldırılarak mobil robotun hareket kabiliyeti arttırılmıştır.

Geliştirilen yazılımdaki kodlar Mini 2440 gömülü sistemde çalışacak şekilde minimize ve optimize edilmiştir. Diferansiyel sürüş sistemine göre hesaplanan hız bilgilerinin mobil robotun motor sürücülerine sinyal olarak gönderilebilmeleri için STM32F4 Discovery Board'dan yararlanılmıştır. Mini 2440 ile STM32F4 arasında sağlanan RS-232 haberleşmesi ile Mini 2440 için geliştirilen yazılımdan STM32F4 cihazına, STM32F4'deki pinlere bağlanan kablolar yardımıyla servo motor sürücülerine hız bilgileri gönderilerek mobil robotun yazılımda hesaplanan yolu takip etmesi sağlanarak, engellerden kaçındığı ve hedefine ulaştığı gözlemlenmiştir.

Daha sonraki çalışmalarda birden fazla robotun hareketli engeller arasındaki yol planlaması, hareketli engellerin olduğu bir hareket ortamının incelenmesi, görüntü işleme kullanılarak yol planlamasının yapılması ve engel geometrisinin dikkate alınması gibi konuların incelenmesinin faydalı olacağı düşünülmektedir.

KAYNAKLAR

[1] **Siegwart, R., and Nourbaksh, I. R.,** 2004. Introduction to Autonomous Mobile Robots, ISBN-13: 978-0262195027, The MIT Press, Cambridge-Massachusetts and London-England, 335p.

[2] **Khatib, O.,** 1986: Real-time obstacle avoidance for manipulators and mobile robots. The International Journal of Robotics Research. Vol. **5**, no. 10, pp. 90-98.

[3] **Url-1** <http://sconkur.pamukkale.edu.tr/zip/files/RoboKolRelease.zip>

[4] **P. E. Hart, N. J. Nilsson, and B.Raphael,** 1968: A formal basis for the heuristic determination of minimum cost paths. IEEE Trans. Syst. Sci. Cybern. Vol. **SSC-4.** no. 2, pp. 100-107.

[5] **Url-2** < http://www.policyalmanac.org/games/aStarTutorial.htm >

[6] **A. Stentz,** 1994. Optimal and efficient path planning for partially-known environments, in Proc. IEEE Int. Conf. Robot. Autom., pp. 3310-3317.

[7] **A. Stentz,** 1993. Optimal and efficient path planning for unknown and dynamic environments, Tech. Rep. CMU-RI-TR-93-20, Carnegie Mellon Univ. Robot. Inst., Pittsburgh, PA.

[8] **A. Stentz,** 1995. The focused D* algorithm for real-time replanning, in Proc. Int. Joint Conf. Artif. Intell., pp. 1652-1659.

[9] **S. X. Yang and M. Meng,** 2000: An efficient neural network approach to dynamic robot motion planning. Neural Netw. Vol. **13**, no. 2, pp. 143-148.

[10] **A. Zelinsky,** 1994: Using path transforms to guide the search for findpath in 2D. Int. J. Robot. Res. Vol. **13**, no. 4, pp. 315-325.

[11] **A.R. Willms and S. X. Yang,** 2006: An efficient dynamic system for real-time robot-path planning. IEEE Trans. Syst., Man, Cybern. B, Cybern. Vol. **36**, no. 4, pp. 755-766.

[12] **A.R. Willms and S. X. Yang,** 2008: Real-time robot path planning via a distance-propogating dynamic system with obstacle clearance. IEEE Trans. Syst., Man, Cybern. B, Cybern. Vol **38**, no. 3, pp. 884-893.

[13] **Yıldız, E.,** 2009. Engelli bir alan içinde otomatik olarak hedefini bulabilen bir mobil robotun tasarımı, imalatı ve hareket algoritmalarının geliştirilmesi, Yüksek Lisans Tezi, Pamukkale Üniversitesi, Denizli.

[14] **Url-3** Penn Engineering, Robotics Homepage. <http://fling-l.seas.upenn.edu>

[15] **Url-4** <http://www.friendlyarm.net/downloads >

ÖZGEÇMİŞ

Ad Soyad: Okan Minnetoğlu

Doğum Yeri ve Tarihi: Denizli 25.06.1987

E-posta Adresi: ominnetoglu@pau.edu.tr

Lisans Üniversite: Pamukkale Üniversitesi, Makine Mühendisliği Bölümü 2009