İSTANBUL TEKNİK ÜNİVERSİTESİ ★ FEN BİLİMLERİ ENSTİTÜSÜ

DEĞİŞKEN BİR ORTAMDA DİNAMİK HYSA YARDIMIYLA GERÇEK ZAMANLI ROBOT YÖNLENDİRME

YÜKSEK LİSANS TEZİ

Volkan KILIÇ

Anabilim Dalı : Elektronik ve Haberleşme Mühendisliği

Programı : Elektronik Mühendisliği

HAZİRAN 2010

İSTANBUL TEKNİK ÜNİVERSİTESİ ★ FEN BİLİMLERİ ENSTİTÜSÜ

DEĞİŞKEN BİR ORTAMDA DİNAMİK HYSA YARDIMIYLA GERÇEK ZAMANLI ROBOT YÖNLENDİRME

YÜKSEK LİSANS TEZİ

Volkan KILIÇ
(504081229)

Tezin Enstitüye Verildiği Tarih : 06 Mayıs 2010
Tezin Savunulduğu Tarih : 10 Haziran 2010

Tez Danışmanı : Doç. Dr. Müştak Erhan YALÇIN (İTÜ)
Diğer Jüri Üyeleri : Prof. Dr. Serdar ÖZOĞUZ (İTÜ)
Prof. Dr. Hakan TEMELTAŞ (İTÜ)

HAZİRAN 2010

Aileme,

ÖNSÖZ

Öncelikle, içindeki bitmeyen enerjisini ve araştırma heyecanını bizlere aktardığı, ilk günden bu yana araştırma ve incelemelerime devam etmem için teşvik ve desteğini hiç eksik etmeyen, tezin hazırlanması sırasında değerli fikirleri ve engin bilgisi ile bana yol gösteren, çok sevgili ve saygıdeğer hocam, tez danışmanım Doç. Dr. Müştak Erhan Yalçın'a teşekkür ederim.

Her türlü bilgiyi paylaşmaktan kaçınmayan sorunlarımı çözmekte büyük yardımları dokunan sevgili arkadaşım Arş. Gör. Ramazan Yeniçeri'ye teşekkürlerimi sunarım.

En zor anlarımda beni motive ederek çalışmalarıma yılmadan devam etmemi sağlayan tezin başarısında önemi büyük olan sevgili Zeynep Köknaroğlu'na teşekkürü yürekten bir borç bilirim.

Son olarak tüm eğitim hayatım boyunca maddi ve manevi destekleri ile her zaman yanımda olan, beni bugünlere getiren biricik aileme ve varlığıyla bana güç veren kardeşim Hakan Kılıç'a teşekkür ederim.

Haziran 2010

Volkan KILIÇ

Elektrik – Elektronik Müh.

İÇİNDEKİLER

Sayfa

KISALTMALAR

HYSA : Hücresel Yapay Sinir Ağları
YSA : Yapay Sinir Ağları
HYSA-EM : Hücresel Yapay Sinir Ağları Evrensel Makinesi
CNN-UM : Cellular Neural Network- Universal Machine
HSA : Hücresel Sinir Ağları
GO-HYSA : Gevşeme Osilatörü tabanlı Hücresel Yapay Sinir Ağı

ŞEKİL LİSTESİ

Sayfa

DEĞİŞKEN BİR ORTAMDA DİNAMİK HYSA YARDIMIYLA GERÇEK ZAMANLI ROBOT YÖNLENDİRME

ÖZET

Bu tez çalışmasında, hücresel yapay sinir ağları en kısa yol bulma probleminin çözümünde kullanılmıştır. Yapılan çalışmalarda gevşeme osilatörü tabanlı hücresel yapay sinir ağı modelinde yürüyen, oto ve spiral dalgalar yayılmış yol bulma probleminin çözümü için yürüyen dalga seçilerek çalışmalar sürdürülmüştür. Geliştirilen ağ modelinin bilgisayar ortamında yapılan benzetimlerinde yol bulma konusunda labirent benzeri ortamda başarılı sonuçlar vermesi üzerine tüm sistemin gerçeklemesine gidilmiştir. Gerçeklemenin yapılması için laboratuar ortamında platform hazırlanmış, ortamın labirente benzemesini sağlamak ve uygulamanın dinamik ortamda çalıştığını göstermek için aynı renge boyanmış tahta bloklar kullanılmıştır. Platform üzerinde kovalama senaryosunu gerçekleştirmek için bluetooth ile haberleşebilen lego robot kullanılmıştır. Tavana yerleştirilen kameradan alınan görüntü işlenerek tasarlanan gevşeme osilatörlü tabanlı hücresel yapay sinir ağında kullanıma uygun hale getirilmiştir. Gerçek ortamda alınan görüntünün ağ da kullanılabilir hale gelmesi için çeşitli görüntü işleme algoritmaları geliştirilmiştir. Ağdan elde edilen yön bilgisini işlemek ve robota göndermek için robot gezinim algoritmaları geliştirilmiştir. Geliştirilen tüm algoritmalarda sistemin gerçek zamanlı ve dinamik bir ortamda çalıştırılabilmesi ön planda tutulmuştur. Tek robotla yapılan çalışmalarda robotun istenilen hedefe ortamdaki ani değişimlere karşı yeni yol haritasını belirleyip ulaşması sağlanmıştır. Robotun duraksamadan hareket etmesi için gerekli iyileştirmeler yapılmış, son gelinen noktada platform üzerinde iki robot kullanılır hale gelinmiş ve takipçi robot kaçan robotu sistemin ürettiği en kısa yol bilgisine göre yakalaması sağlanmıştır.

REAL TIME ROBOT NAVIGATION IN VOLATILE ENVIRONMENT WITH DYNAMIC CNNA

SUMMARY

In this thesis, cellular neural network is used to solve finding the shortest path problem. Travelling, auto and spiral waves are propagated on the relaxation oscillator based cellular neural network and travelling waves are chosen in order to solve path finding problem in that studies. After developed network model is successful in simulation about path finding on the maze-like environment, whole system is going to be implemented. For the implementation, platform is prepared in a laboratory, in order to make the environment look like maze and ensure the application is working in a dynamic environment, same color painted wooden blocks used. To perform the chase scenarios on the platform, lego robots are used which can communicate via Bluetooth. Image which is taken from camera placed on the ceiling is processed to be able to use on the relaxation oscillator cellular neural network design. Several image processing algorithms are developed to make the image taken from real environment usable in the network. Robot navigation algorithms are developed to process the direction data received from the network and send to robot. It is intended to make the system work in real time and dynamic environment in all developed algorithms. In studies with single robot, it is provided that robot finds new path then reach the target point against the sudden changes in the environment. System is optimized to make the robot move nonstop and finally, two robots can be used on the platform and it is provided that tracker robot tries to catch the other robot using the shortest path information provided by the system.

1. GİRİŞ

Hücresel Yapay Sinir Ağ mimarisi, Chua ve yang [1] tarafından tanıtılmasından sonra birçok mühendislik uygulamasına konu olmuştur. İki-boyutlu ızgara biçiminde bir bağlantı yapısına sahip HYSA, mimarisinin doğal bir sonucu olarak ana uygulama alanı olarak görüntü işlemede kullanılmıştır [2, 3]. Çalışmaların ilerletilmesiyle insan retinasının HYSA'da modellenmesi başarılmış, böylelikle HYSA'lar biyolojik temel kazanmıştır [4].

Doğrusal olmayan uzay-zaman dalgası üreten HYSA modeli Yalçın tarafından 2008 yılında önerilmiştir [5]. Bu model gevşeme osilatörlerinden birleştirilmesiyle oluşturulmuş ve tezde sunulan çalışmanın temelinde GO-HYSA modeli bulunmaktadır. GO-HYSA modeli üzerinde aktif dalgalar üretilmesi sağlanmış [6, 7] ve bu dalgaların kullanarak en kısa yol bulma problemine çözüm getirildiği benzetim sonuçlarıyla görülmüştür [7, 8]. Bu çalışmada, benzetimden elde edilen başarı sonuçlara dayanarak GO-HYSA ile en kısa yol bulma olayının gerçeklenmesi yapılmıştır.

En kısa yolu bulma konusunda aktif dalgaları kullanarak yapılmış bir çok çalışma mevcuttur[9-16]. Bu çalışmalarda aktif dalgaların gözlendiği kimyasal ortamlar ile kurulan işlemciler, HYSA'nın analojik devre gerçeklemeleri [17] ve yeniden konfigüre edilebilir sayısal cihazlar, örneğin sahada programlanabilir kapı dizileri (FPGA) gibi donanımlar kullanılmıştır [18, 19]. Bu tez çalışmasında ise GO-HYSA modeli, görüntü işleme ve robot takip senaryosunun tamamı Matlab programı altında çalıştırılmıştır [20, 21].

GO-HYSA üzerinde yapılan çalışmalarda üç çeşit dalga tipi oluşturulmuştur. Bunlar yürüyen dalgalar, oto dalgalar ve spiral dalgalardır. Yapılan gözlemler sonucunda yol bulma probleminde yürüyen dalga kullanımının diğer dalga türlerine göre daha basit ve etkili olduğu görülmüştür. Yürüyen dalga, tek bir noktadan başlayarak yayılmakta ve yayılım tüm arenayı kaplamadan ikinci bir dalga yayılmamaktadır.

GO-HYSA ile yön bulma olayının gerçeklenmesi için laboratuar ortamında platform hazırlanmış, aynı renge boyalı tahtalardan engeller hazırlanmıştır. Bluetooth ile

haberleşebilen robot seçilmiştir. Platformun tepeden çekilen resmi görüntü işleme metoduyla işlenerek tahtalar engel olarak tespit edilip ağda bu noktalara karşılık gelen yerler pasif hücre olarak atanmıştır. Yine görüntü işlemeyle kaçan ve kovalayan robotların koordinatları tespit edilip, kaçan robotun koordinatı ağa kaynak hücre olarak atanmıştır. Ağın parametreleri yürüyen dalga üretecek şekilde ayarlanarak, yön bulma probleminin gerçeklenmesi sağlanmıştır. Robot gezinim algoritmaları ve yürüyen dalganın yayılım algoritması geliştirilerek tüm sistemin dinamik ortamda düzgün bir şekilde çalışması sağlanmıştır [20, 21].

2. HÜCRESEL YAPAY SİNİR AĞLARINA GENEL BAKIŞ

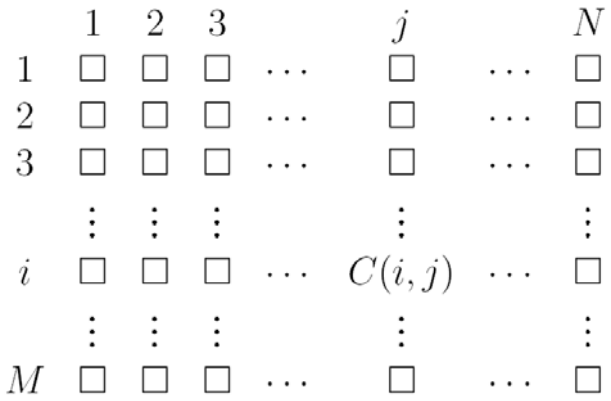
Leon O. Chua ve Lin Yann 1988 de yayınladıkları bildiriyle tasarladıkları hücresel yapay sinir ağı mimarisini tanıtması bilim dünyasının ilgisini üzerine çekmiştir [1]. Ortaya koydukları mimari, sinir sistemindeki hücrelerin diğer hücrelerle olan ilişkisinden esinlenmiş ve hücreler birbirine yerel olarak bağlanmıştır. Hücresel yapay sinir ağları(HYSA), hücrelerden meydana gelmiş ve ağdaki her bir hücre en yakınındaki hücre ile etkileşim ve komşuluk ilişkisi içerisinde bulunan dinamik yapay sinir ağ(YSA) modelidir. HYSA'ların ayırıcı en önemli özelliği, YSA' da yapılan genel(global) bağlantıların aksine bağlantıların yerel olarak yapılmış olmasıdır. Örnek olarak, ileri beslemeli bir sinir ağında, katmanın bir hücresi diğer katmanların tüm hücrelere bağlıdır veya hopfield ağlarında hücrelerin her biri diğer tüm hücrelere bağlıdır. HYSA'ların bir özelliği de ağırlıklar sistemin dinamikliğini kararlaştırılmasında kullanılırken, YSA da ağırlıklar sistemin önceki durumları veya geri besleme bilgilerini öğrenmede kullanılır.

Chua ve Roska nın 1993 te yayınladığı çalışmasıyla HYSA paradigmasını ortaya konmuştur [3]. Yine aynı yıl yayınladıkları bildiriyle ilk defa algoritmik olarak programlanabilen analog HYSA işlemcisini tanıttılar. Bu çalışmayla HYSA tabanlı işlemcilerin üretilebileceğini kanıtlamış oldular ve araştırmacılarına kendi HYSA teorilerini test edebilecekleri fiziksel bir ortam sunmuş oldular. HYSA-EM (CNN-UM) olarak adlandırılan işlemci analog ve lojik işlem yeteneğinden dolayı analogic işlemci olarak nitelendirilmiştir.

HYSA, yüksek ve alçak geçirgen süzgeç yapımında, morfolojik işlemlerde, diferansiyel denklem çözümünün gerekli olduğu ısı kaybı ve dalga yayılımı hesaplamalarında kullanılabilmektedir. HYSA, ağ mimarisinin özelliklerinden dolayı en geniş kullanım alanı görüntü işlemede bulmuştur. Retina modellemesi ve biyonik göz gibi uygulamaları da bulunmaktadır. Yapılan çalışmalar, HYSA'nın gelecekte daha geniş bir uygulama alanında kullanılacağını göstermektedir.

2.1 Hücresel Yapay Sinir Ağlarının Mimari Yapısı

Hücresel yapay sinir ağları bilginin giriş, eşik ve başlangıç durumu olarak adlandırılan üç adet bağımsız değişken kullanılarak, hücre (cell) adı verilen doğrusal-olmayan (nonlinear) dinamik sistemler tarafından işlendiği bir ağ yapısıdır. Standart bir HYSA yapısı, C(i, j) hücrelerinin oluşturduğu M x N boylarında iki boyutlu bir dizi olarak tanımlanabilir. Buradaki (i, j) kartezyen koordinat indisleri i=1,2,3,…,M ve j=1,2,3,…,N şeklindedir. Standart HYSA'nın yapısı Şekil 2.1'de verilmiştir.



Şekil 2.1: Standart HYSA hücre yapısı

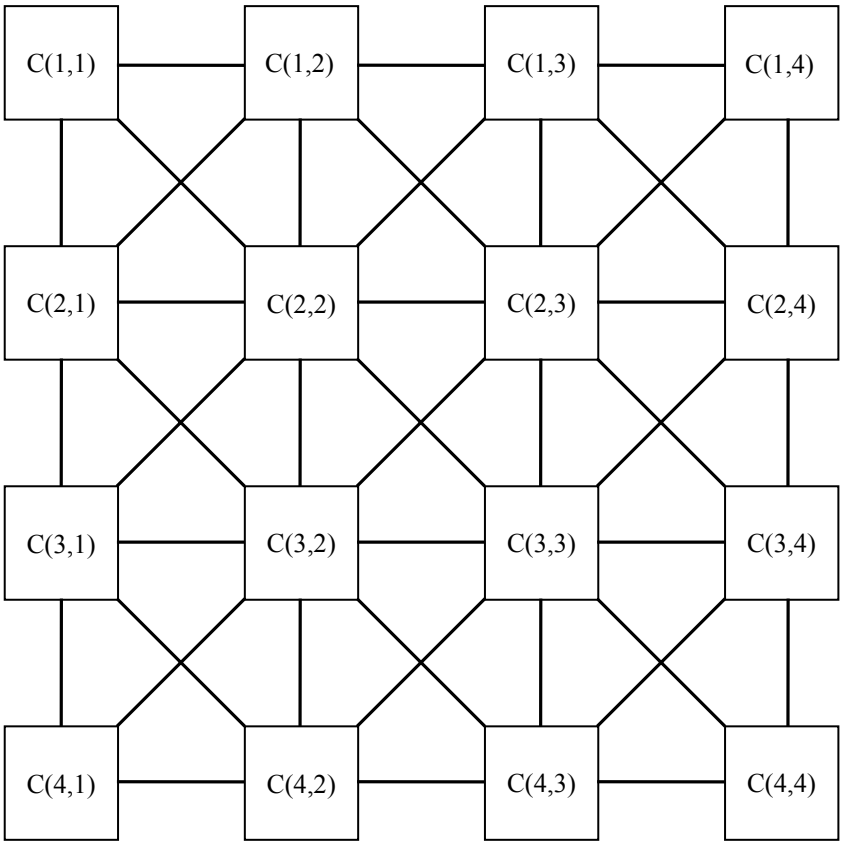
Chua ve Yang tarafından 1988 de ortaya atılan, her bir karenin temel birim olan hücreyi simgelediği 4x4 boyutlu HYSA'nın temel mimarisi Şekil 2.2'de gösterilmektedir. Bu HYSA mimarisinde her hücre yalnızca komşu hücreleriyle etkileşim içindedir. Komşu hücreler birbirlerini doğrudan, diğer hücreleri komşuları üzerinden dolaylı olarak etkilerler.

HYSA, C(i, j) olarak tanımlanan her bir hücrenin, r yarıçaplı komşuluk dairesi $S_r(i, j)$ içerisinde bulunan C_{kl} hücreleri ile olan bağlantılarını (coupling laws) gösteren bir ağ yapısını göstermektedir [22].

Etki alanı $S_r(i, j)$,C(i, j) hücresinin kendisinin r komşuluğuna kadar olan hücrelerinin oluşturduğu bir kümedir şu şekilde tanımlanır:

$$S_r(i, j)=\{ C(k,l) \mid \max\{|k-i|, |l-j|\} \le r, \quad 1 \le k \le M;\ 1 \le l \le N \} \qquad \textbf{(2.1)}$$

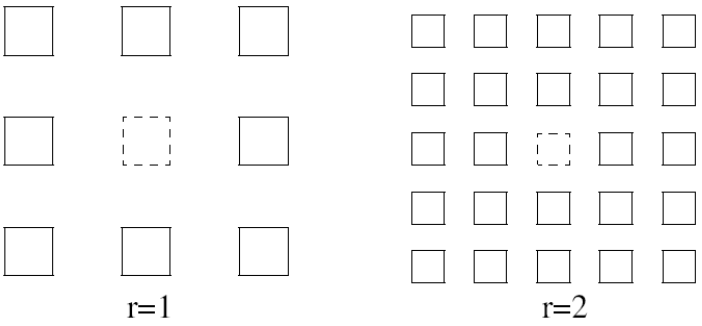
Burada r pozitif bir tamsayıdır.



Şekil 2.2: İki boyutlu 4x4 hücresel sinir ağı modeli

Aynı hücrenin(merkezde ve kesikli çizgiyle gösterilmiştir), iki farklı komşuluğunu r=1 ve r=2 için

Şekil 2.3'de gösterilmiştir. Komşuluk sistemi bir simetri özelliği göstermektedir. Öyle ki, HYSA içerisindeki bütün $C(i, j)$ ve $C(k,l)$ için $C(i, j) \in S_r(k,$



Şekil 2.3: r = 1 ve r = 2 için C(i , j) hücresinin komşulukları

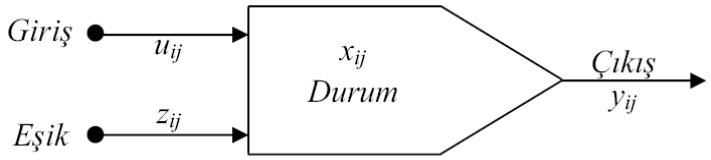
Buraya kadar anlatılan HYSA'nın özelliklerine bakarak, HYSA'nın belirli kurallara göre işlem yapan bölgesel olarak birbirine bağlı (locally-coupled) hücreler topluluğudur. Bu topluluk içerisindeki her bir dinamik sistem, giriş (input), çıkış

(output), ve durum (state) değerine sahiptir[22]. Genelleştirilmiş bir HYSA denklemi (2.2) de verilmiştir.

$$\frac{dx}{dt} = -x_{ij} + \sum_{C(k,l)\in Sr(i,j)} A_{i,j;k,l} y_{k,l} + \sum_{C(k,l)\in Sr(i,j)} B_{i,j;k,l} u_{k,l} + z_{i,j}$$

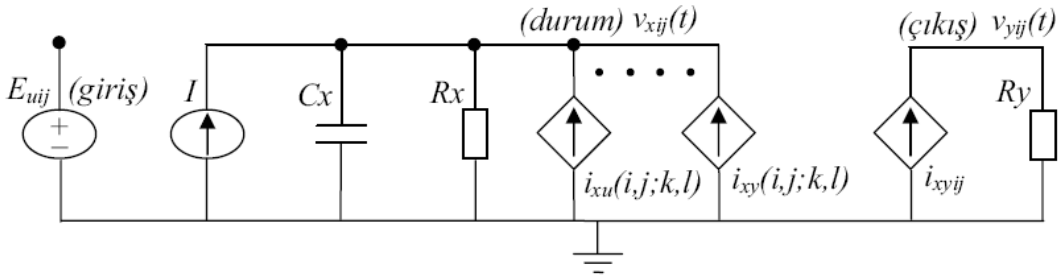
$$y_{ij} = f(x_{ij}) = \frac{1}{2}\left|x_{ij}+1\right| - \frac{1}{2}\left|x_{ij}+1\right|; \quad i=1,2,\ldots,M,\ j=1,2,\ldots,N. \tag{2.2}$$

(2.2)' de verilen denklemde x_{ij}, y_{ij}, u_{ij} ve z_{ij} sırasıyla bir hücrenin durum, çıkış, giriş ve eşik ifadelerini $Sr(i,j)$ hücrenin r yarıçaplı komşuluk kümesini oluştururken $A_{i,j;k,l}$ ve $B_{i,j;k,l}$ ise hücrenin sinaptik ağırlıklarını ifade etmektedir. Ayrıca z_{ij} zamanla değişmeyen bir akım sabitini temsil ederken x_{ij}, y_{ij}, u_{ij}'nin zamana bağımlı fonksiyonlar oldukları kabul edilmektedir. Bir HSA hücresinin blok şeması Şekil 2.4 de gösterilmiştir. Görüldüğü üzere giriş hücresi giriş(input), eşik(threshold), durum(state) ve bunlara bağlı olarak da çıkış(output) değerine sahiptir.



Şekil 2.4: HYSA hücresinin blok olarak gösterimi

HYSA içerisindeki tipik bir Cij hücresinin devre yapısı Şekil 2.5' de verilmiştir. Cij' nin düğüm gerilimi V_{xij} hücre durumu olarak adlandırılır. Düğüm gerilimi V_{uij}, Cij'nin girişi olarak adlandırılır ve büyüklüğü bir ya da daha az olan bir sabit olduğu kabul edilir. Düğüm gerilimi V_{yij} çıkış olarak adlandırılır.[22]



Şekil 2.5: HYSA hücresinin devre şeması

Her bir C_{ij} hücresi Şekil 2.5'ten de görüldüğü üzere bir tane bağımsız gerilim kaynağı E_{ij}, bir tane bağımsız akım kaynağı I, bir tane kapasitör C_x, iki tane doğrusal

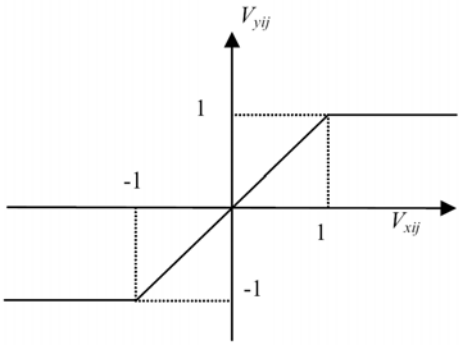
direnç R_x ve R_y ile her bir komşu hücre C_{kl}'nin giriş gerilimi V_{ukl} ve çıkış gerilimi V_{ykl}'nin geri beslemesi doğrultusunda komşu hücrelerden kuplajlanan gerilim kontrollü akım kaynakları içermektedir. $I_{xy(i,\,j;k,l)}$ ve $I_{xu(i,\,j;k,l)}$ doğrusal gerilim kontrollü akım kaynakları içermektedir ve (2.3)'de gösterildiği gibi tanımlanmaktadır.

$$I_{xy}(i,\,j;k,l) = A(i,\,j;k,l)V_{ykl}$$
$$I_{xu}(i,\,j;k,l) = B(i,\,j;k,l)V_{ukl} \qquad \textbf{(2.3)}$$

Her bir hücredeki doğrusal olmayan tek eleman, bir parçalı-doğrusal (piecewise-linear) gerilim kontrollü akım kaynağıdır ve (2.4)'de gösterildiği gibi tanımlanmaktadır.

$$I_{yx} = \frac{1}{R_y} f(V_{xij}) \qquad \textbf{(2.4)}$$

Şekil 2.6'da fonksiyonun parçalı-doğrusal yapısı ile hücre çıkışı gösterilmektedir.



Şekil 2.6: Parçalı-doğrusal karakteristiğe sahip hücre çıkışı

Şekil 2.5 de verilen devre için hücre eşitliklerini özetlemek gerekirse, HYSA hücre denklemleri için parametre kabulleri aşağıdaki gibi yapılabilir:

Durum denklemi:

$$C\frac{dV_{xij}(t)}{dt} = -\frac{1}{Rx}V_{xij}(t) + \sum_{C(k,l)\in N_r(i,j)} A(i,\,j;k,l)V_{ykl}(t) + \sum_{C(k,l)\in N_r(i,j)} B(i,\,j;k,l)V_{ukl}(t) + I, \qquad \textbf{(2.5)}$$
$$1 \le i \le M;\;\; 1 \le j \le N.$$

Çıkış denklemi:

$$V_{yij}(t) = \frac{1}{2}\left(|V_{xij}(t)+1| - |V_{xij}(t)-1|\right),\;\; 1 \le i \le M;\;\; 1 \le j \le N. \qquad \textbf{(2.6)}$$

Giriş denklemi:

$$V_{uij} = E_{ij}\,,\ 1 \le i \le M;\ 1 \le j \le N. \tag{2.7}$$

Sınır şartları:

$$\begin{aligned} &\left|Vxij(0)\right| \le 1\,, \qquad 1 \le i \le M;\ 1 \le j \le N. \\ &\left|V_{uij}\right| \le 1, \qquad\quad 1 \le i \le M;\ 1 \le j \le N. \end{aligned} \tag{2.8}$$

Parametre kabulleri:

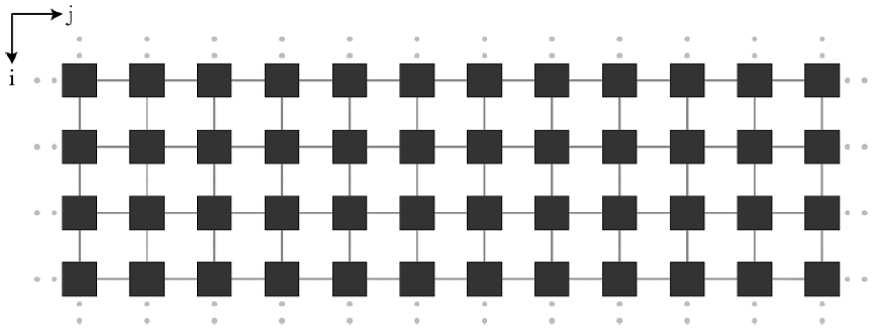
$$\begin{aligned} &A(i,j;k,l) = A(k,l;i,j),, \qquad 1 \le i \le M;\ 1 \le j \le N. \\ &C > 0, R_x > 0. \end{aligned} \tag{2.9}$$

2.2 Gevşeme Osilatörlü Hücresel Yapay Sinir Ağları

Bir önceki bölümde HYSA'nın mimari yapısı ve dayandığı bilimsel temeller açıklanmıştır. Bu bölümde Yalçın'ın 2008'de önerdiği elektronik gerçekleme açısından en uygun olarak tanıtılan doğrusal olmayan uzay-zaman dalgası üreten kare dalga osilatörü kullanarak modellenen sinir hücresi modeli açıklanacaktır [5]. Modellemenin sonrasında ağa ait bir hücrenin benzetimi 2.3.1, 128x128 lik bir HYSA'nın benzetimi 2.3.2'de yapılmıştır. Burada yapılan çalışmalara dayanarak otodalgalar, spiral dalgalar ve yürüyen dalgaların yayılımı gösterilmiştir.

Bu çalışmada kare dalga osilatörü kullanılmıştır ve bunun literatürdeki adı Gevşeme osilatörüdür. Yalçının önerdiği ağ modeli esasen dört komşusu ile bağlantılı Gevşeme osilatörlerinin Şekil 2.7'deki mimari ile birleştirilmesi ile oluşturulmuştur.

Yalçın, bir HYSA gerçeklemesi olan ACE16k tümdevresi ile yaptığı çalışmalar sonucunda [23] bahsedilen osilasyon davranışını sergileyecek hücresel modelin (2.10)'daki gibi olduğunu bulmuştur.



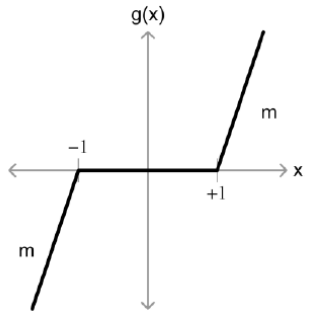
Şekil 2.7: İki boyutlu hücresel olmayan doğrusal ağ

$$\frac{dx}{dt} = \alpha x + \beta y - g(x)$$
$$\frac{dy}{dt} = (x - y) \qquad \textbf{(2.10)}$$

Denklemdeki x ve y durum değişkeni α, β ve ε reel değerli parametrelerdir. Ayrıca modelde doğrusal olmayan davranışın kaynağı olan g(x) fonksiyonu da saptanmış ve (2.11) 'de ifade edilmiştir.

$$g(x) = \begin{cases} m(x+1) & ; \quad x < -1 \\ 0 & ; \quad |x| \le 1 \\ m(x-1) & ; \quad x > 1 \end{cases} \qquad \textbf{(2.11)}$$

Şekil 2.8'de g fonksiyonunun x'e göre grafiksel ifadesi gösterilmiştir.



Şekil 2.8: Doğrusallığı bozan g(x) fonksiyonu

Tek başına salınım yapan hücrenin x durum değişkeni, zaman içerisinde kare dalga işareti oluşturmaktadır. Bu osilatörlerden oluşturulacak HYSA'nın hücresel modeli için yine [5]' den yararlanarak model denklemlerine ulaşılmıştır. Yalçın'ın önerdiği modelde değişikliğe gidilmeye ihtiyaç duyulmuştur. Bunlardan ilki g fonksiyonunun eğiminin negatif yapılarak g'nin x durumuna ait diferansiyel denklemde toplanan terim haline getirilmesidir. Diğer değişiklik ise y durumuna ait diferansiyel denklemde y'nin ağırlığının x'in ağırlığından bağımsız hale getirilmesi, yani –ε yerine σ ağırlık parametresinin kullanılması olmuştur. Yapılan değişiklerden sonra diferansiyel denklemin son hali (2.12) da verilmiştir.

$$\frac{dx}{dt} = \alpha x_{i,j} + \beta y_{i,j} + g(x_{i,j}) + I_{i,j} + u_{i,j}$$
$$\frac{dy}{dt} = \varepsilon(x_{i,j} - y_{i,j}) \qquad \textbf{(2.12)}$$

Gerek sayısal devre gerçeklemelerinin gerekse bilgisayar benzetimlerinin yapılabilmesi için (2.12)'da verilen model denklemleri ileri Eular integrasyon metodu ile zamanda ayrık hale getirilmiştir. Böylelikle bu çalışmada kullanılacak hücreye ait model denklemleri (2.13)'daki gibi olmuştur.

$$x_{i,j}[k+1] = x_{i,j}[k] + T\left(\alpha x_{i,j}[k] + \beta y_{i,j}[k] + g\left(x_{i,j}[k]\right) + I_{i,j}[k] + u_{i,j}\right)$$
$$y_{i,j}[k+1] = y_{i,j}[k] + T\left(\varepsilon x_{i,j}[k] - \sigma y_{i,j}[k]\right) \qquad \textbf{(2.13)}$$

Buradaki T integrasyon adımıdır. Doğrusallığı bozan g(x) fonksiyonu da (2.14)'de verilmiştir.

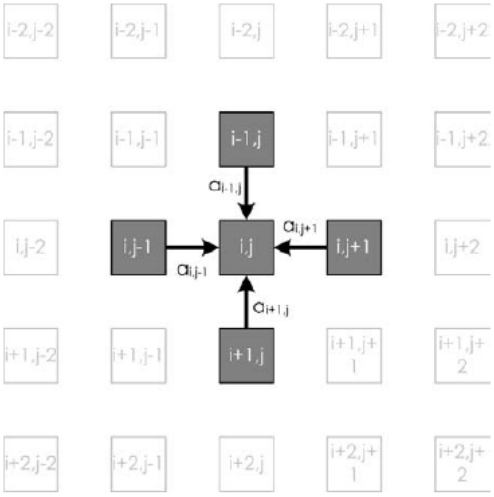
$$g\left(x_{i,j}[k]\right) = \begin{cases} m\left(x_{i,j}[k] + \lambda\right) & ; \quad x < -\lambda \\ 0 & ; \quad |x| \le \lambda \\ m\left(x_{i,j}[k] - \lambda\right) & ; \quad x > \lambda \end{cases} \qquad \textbf{(2.14)}$$

Şekil 2.8' de görülen g(x) fonksiyonunun doğrusal olduğu aralıkları belirleyen değer (2.14)'de parametrik hale getirilmiş ve λ ile gösterilmiştir.

I(.) fonksiyonu ağın oluşmasını sağlamaktadır. I, her bir hücreye komşularının x durum değişkenlerinin ağırlıklandırılmış toplamını taşır. Komşuluk terimi ismiyle anılacak olan I, (2.15) eşitliği ile tanımlanmaktadır. Hücresel modelin ayrık zamanlı denklemleri incelendiğinde her bir hücreye ait durum değişkenlerinin i ve j indisleri ile belirtilmiş olduğu görülmektedir.

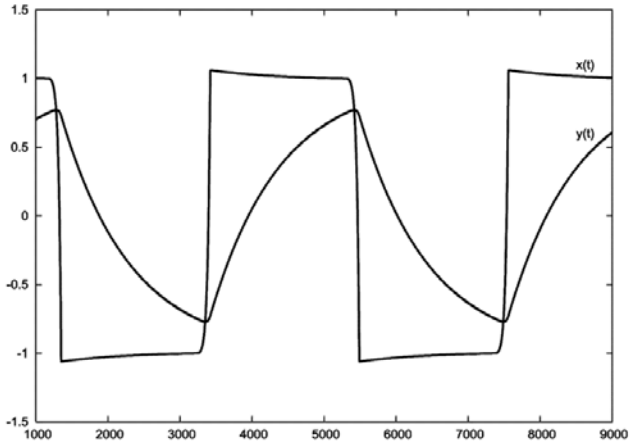
$$I_{i,j}[k] = a_{i,j+1}x_{i,j+1}[k] + a_{i-1,j}x_{i-1,j}[k] + a_{i,j-1}x_{i,j-1}[k] + a_{i+1,j}x_{i+1,j}[k] \qquad \textbf{(2.15)}$$

İki boyutlu (MxN) hücresel yapay sinir ağları M satır ve N sütundan oluşmakta, hücrelerden her birinin yeri bir sütun ve bir satır değeri ile belirtilmektedir [1]. Tez boyunca i indisi hücrenin bulunduğu satırı, j indisi de hücrenin bulunduğu sütunu belirtecektir. (2.13)'da bu indisler kullanılmış, (2.15)'de komşu hücreler yine bu indislerle ifade edilmiştir. İncelendiğinde, komşuluk terimi I, referans alınan hücrenin HYSA'daki doğu, kuzey, batı ve güney komşularının referans hücreye bağlamaktadır. Şekil 2.9'de bu komşuluk bağlantısı gösterilmiştir. Modelde, sürekli giriş *u* ile ifade edilir. Kurulan HYSA'ndan beklenen davranış çeşitli aktif dalgaların üretilmesidir. Yapılan çalışmalarda bu dalgaların kaynağının u giriş işareti ile belirlenebildiği gösterilmiştir.



Şekil 2.9: Ağ üzerindeki indislerin ve komşuluk terimi I'nın gösterimi.

Yine de genel HYSA modelinde bulunan giriş işareti terimi, bu çalışmada kullanılan son modelde de korunmuştur. Yukarıdaki ifadeler ile kurulan hücresel modelin, HYSA göz önünde bulundurulduğunda bir eksiğinin bulunduğu fark edilir. Mevcut ifade HYSA'daki tüm hücrelerin dört komşuluğunda gerçek hücrelerin var olduğunu ve bu hücrelerden değerler aldığını kabul etmektedir. Fakat gerçekte HYSA'nın kenar ve köşelerinde yerleşik hücrelerin bir ya da iki (köşedekiler için) komşusu bulunmamaktadır. Bu hücreler için belirlenmesi gereken sınır koşulları ileride anlatılacak benzetim ve gerçeklemelerde ayrıca açıklanmıştır. Özellikle devre gerçeklemelerinde sınır koşulları için farklı çözümler uygulanmıştır. Örneğin Altbölüm 2.3.1 'de tek hücrenin benzetimi yapılırken, bu hücrenin tüm komşuları için sınır koşulları kullanılmıştır. Temel alınan modelinde Yalçın, tek hücreye ait x ve y durum değişkenlerinin zaman içinde Şekil 2.10'deki gibi değiştiğini göstermektedir.



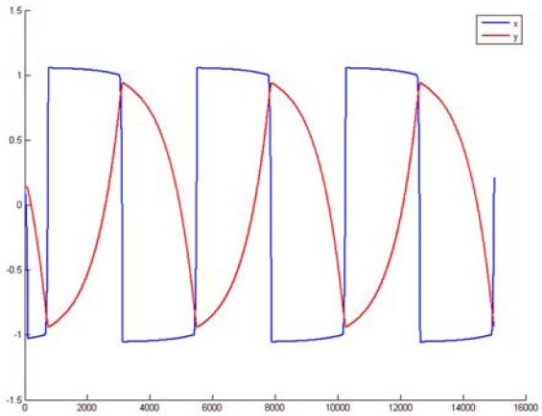
Şekil 2.10: Hücrenin x ve y durumlarının zamanla değişimi

2.3 Hücre benzetimleri

HYSA'nın üreteceği dalgaların dinamiği kullanılan modelle ve model parametreleriyle doğrudan ilişkilidir. Matematiksel temelin oluşturulmasından sonra çalışmaya benzetimler ile devam etmenin bir sebebi kaynaklarda verilen parametrelerin test edilmesi ve uygun parametre aralıklarının araştırılabilmesidir. Hücreye ait matematiksel modelin benzetimi Matlab ortamında yapılmıştır. Tez boyunca R2009a sürümü kullanılmıştır. Benzetimlerde çift duyarlı kayan nokta aritmetiği ile işlemler gerçekleştirilmiştir. Tezde tek hücreye ve 128x128 hücrelik ağa ait benzetim sonuçları sırasıyla Altbölüm 2.3.1 ve Altbölüm 2.3.2'de verilmiştir.

2.3.1 Tek Hücre benzetimi

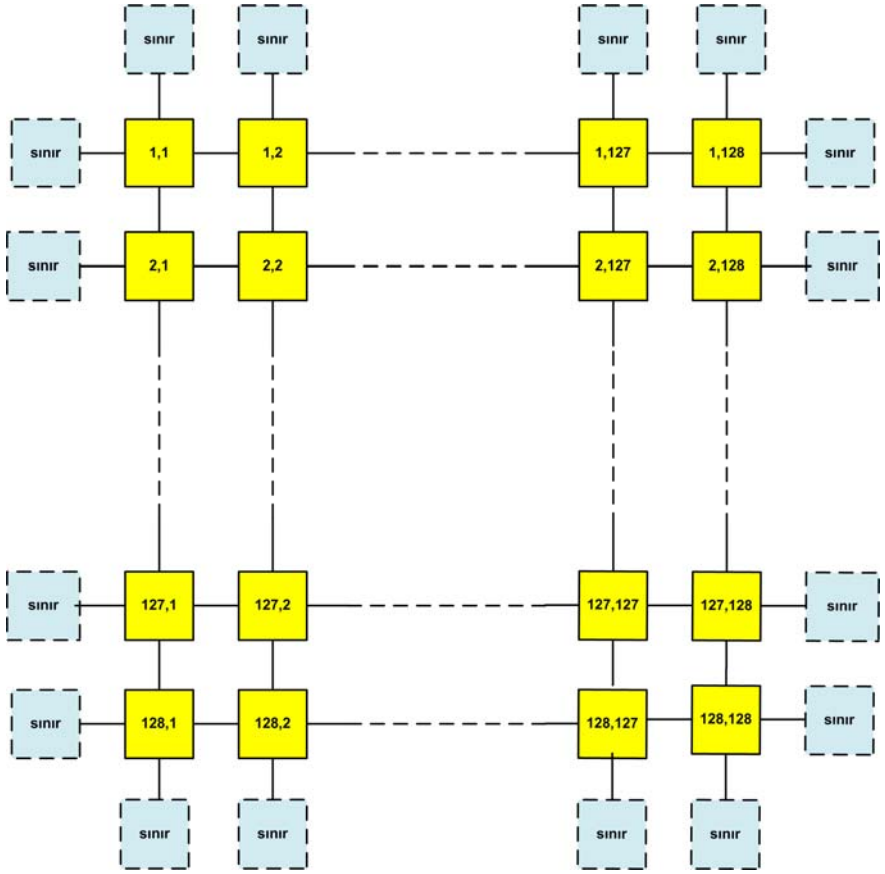
Yapılan tek hücre benzetimi ile *x* ve *y*'nin zamanla değişiminin, Şekil 2.10'da gösterilen davranışı sergilenmesi hedeflenmiştir. Tek hücrenin benzetiminde dört komşudan gelen bağlantı değerleri 0 yapılmıştır. Dolayısı ile her ne kadar 0.1 olarak verilse de, komşuluk ağırlıklarının hücre dinamiğinde etkisi yoktur. Hücreye ait durum değişkenlerinin başlangıç değeri 0.125 verilmiştir. Yapılan benzetim sonucunda *x* ve *y* durumlarının zamanla değişimi Şekil 2.11'de verilen grafiğe çizdirilmiştir. Şekil 2.10 ve Şekil 2.11'deki grafikler beklendiği gibi davranışların örtüştüğünü göstermektedir. Kare dalga işaretinin frekansı model parametreleri ile değişmektedir. Benzetim, sadece *x* ve *y* durum değişkenlerinin başlangıç değerleri değiştirilerek koşturulduğunda işaretin fazının kaydığı görülmüştür. Model, birbirine bağlanmamış komşu hücrelerin farklı başlangıç koşulları ile koşturulduğunda, farklı fazlarda işaretler üreteceklerini ve işaretlerin *i*,*j* indisli uzayda da bir uzay dalgasını oluşturabileceğini göstermektedir. Hücre bazında zamanla değişen bir dalga, düzgün yerleştirilmiş hücreler uzayında, uzayda yol alan dalgaları oluşturmaktadır.



Şekil 2.11: Tek hücrenin benzetim sonucunda elde edilen x ve y değişkenleri.

2.3.2 Gevşeme osilatörlerinden oluşmuş ağ üzerinde doğrusal olmayan uzay zaman dalgaları

Tek hücrenin benzetiminden istenilen sonuçların alınmasının ardından, 128 x 128 lik bir ağ tasarlanmış ve bu ağ üzerinde uzay-zaman dalgaları yayılmaya çalışılmıştır. Tek hücrenin benzetim kodu 128x128 hücre grubu için tekrar ele alınmış ve bir ağ oluşturulmuştur. Hücreler birbirine uygun komşuluklarda bağlanmıştır. Ağ oluşturulması ile sınır koşullarının belirlenmesine gerek duyulmuştur. Ağın kenarlarında bulunan hücrelerin komşularına bağlanması gereken birer girişleri, köşelerindeki hücrelerin de ikişer girişleri gerçek komşulara bağlanmamıştır. Sınır koşulu, bu hücrelerin eksik komşularına ait girişlerine hangi sabit değerin sürekli uygulanacağını belirtir. Benzetimde sınır koşulu 0 seçilmiştir. Bahsedilen 128x128 lik HYSA yapısı Şekil 2.12'de verilmiştir.



Şekil 2.12: 128x128 HYSA yapısı

Ağ oluşumuyla komşuluk bağlantılarının ağırlıkları olan a parametreleri de önem kazanmıştır. Bu ağırlıklara 0 değeri verilirse tüm hücreler birbiri ile bağımsız hale gelir ve her biri tek hücre benzetimindeki gibi salınır. Ağırlıkların sıfırdan farklı bir değer alması ile hücreler osilasyona devam eder fakat aralarında tıpkı farklı başlangıç

koşulları ile başlatılmış gibi faz farkı oluşur. Faz farkının oluşması ağ üzerinde uzaysal dalgaların oluşmasına sebep olan temel olaydır. (2.13)'deki α, β, ε, σ; ağırlıkları; (2.15)'de verilen a ağırlıkları; g(.) fonksiyonunu tanımlayan μ ve λ parametreleri ile farklı başlangıç koşulları ve giriş işaretleri uygulanarak ağın çeşitli uzay-zaman dalgalarını üretmesi ve yayması sağlanabilmektedir. Ağ modelinin bilgisayar benzetimi gerçekleştirilmiş ve uzay-zaman dalga formları olan oto dalgalar, yürüyen dalgalar, spiral dalgalar gözlenmiştir. 128x128 lik ağın bilgisayar benzetimi ile oto dalga üretmesi Şekil A.1' de gösterilmiştir. Oto dalganın üretimi denklem (2.13), (2.14) ve (2.15) te kullanılan parametre ve ağırlıkların $\alpha = 1$, $\beta=-6$, $\varepsilon=0.1$, $\sigma=-0.1$, $a_{i,j+1}=0.2$, $a_{i,j-1}=0.2$, $a_{i+1,j}=0.2$ ve $a_{i-1,j}=0.2$, $\mu=-20$, $\lambda=1$, $u_{kaynak}= -0.1$ atanmasıyla gerçeklenmiştir.Yine denklem (2.13), (2.14) ve (2.15) te kullanılan parametre ve ağırlıkların $\alpha = 1.5$, $\beta=-3.9$, $\varepsilon=1$, $\sigma=-1.1$, $a_{i,j+1}=0.8$, $a_{i,j-1}=0.8$, $a_{i+1,j}=0.8$ ve $a_{i-1,j}=0.8$, $\mu=-20$, $\lambda=1$ atanmasıyla Şekil A.2'de gösterilen spiral dalgalar üretilmiştir. Spiral dalganın üretiminde oto dalganın üretiminden farklı olarak sadece ağırlıklar ve parametreler değiştirilmemiş ayrıca özel başlangıç koşulları atanmıştır. Ağda yürüyen dalganın üretilmesi de gerçeklenmiş ve Şekil A.3 de gösterilmiştir. Bunun için parametre ve ağırlıklara spiral dalgalarla aynı değerler verilmiştir. Farklı davranışın sebebi farklı x başlangıç koşulları ve u giriş işareti uygulanmasıdır. Ağın en önemli bir özelliği de x durumunun 0'a sabitlenebilmesidir. Bu sayede ağ üzerinde sabitlenmiş hücreler tanımlanabilmektedir. Doğrusal olmayan uzay-zaman dalgalarının bu sabit kılınan hücrelerin etrafından dolaştığı ya da aralarındaki geçitlerden geçtiği gözlenir. Şekil A.4'da bazı hücrelerin değerleri sabit tutularak engelli ortam diye de tabir edilen labirent benzeri bir ortamda yürüyen dalganın yayılımı gösterilmektedir. Bu çalışmadaki uzay-zaman dalgalarının diğer klasik uzay-zaman dalgalarından farklı olarak şu iki özelliğe sahiptir:

- Çeperler birbiri ile karşılaşınca girişim olmaz ve çeperler birbirini söndürür.
- Çeper engele(sabit hücrelere) çarpınca yansımaz ve engel çeperi söndürür.

Yansımama ve girişim yapmama uzay zaman dalgalarının tanımlayıcı özellikleridir. Bu sayede uzay-zaman dalgalarının farklı alanlarda uygulanabilirliği düşünülmeye başlanmıştır.

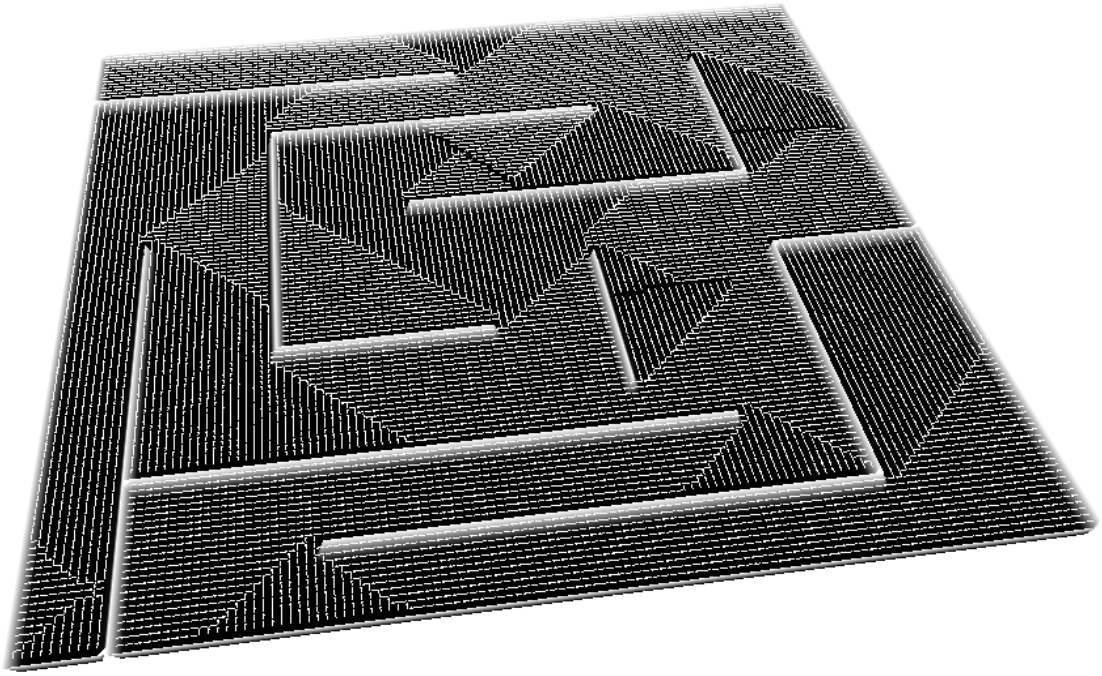
3. GEVŞEME OSİLATÖRLÜ HYSA'DA DİFÜZYONA DAYALI YOL BULMA ALGORİTMASI

Bir önceki bölümde ağ üzerinde oto dalga, spiral dalga ve yürüyen dalganın yayılımı gerçekleştirilmiştir. Yayılan üç dalgalanın incelenmesinden sonra yön bulma probleminin çözümünde yürüyen dalganın kullanımına karar verilmiştir ve en kısa yol haritasını elde edebilmek için difüzyona dayalı bir yol bulma algoritması geliştirilmiştir [7].

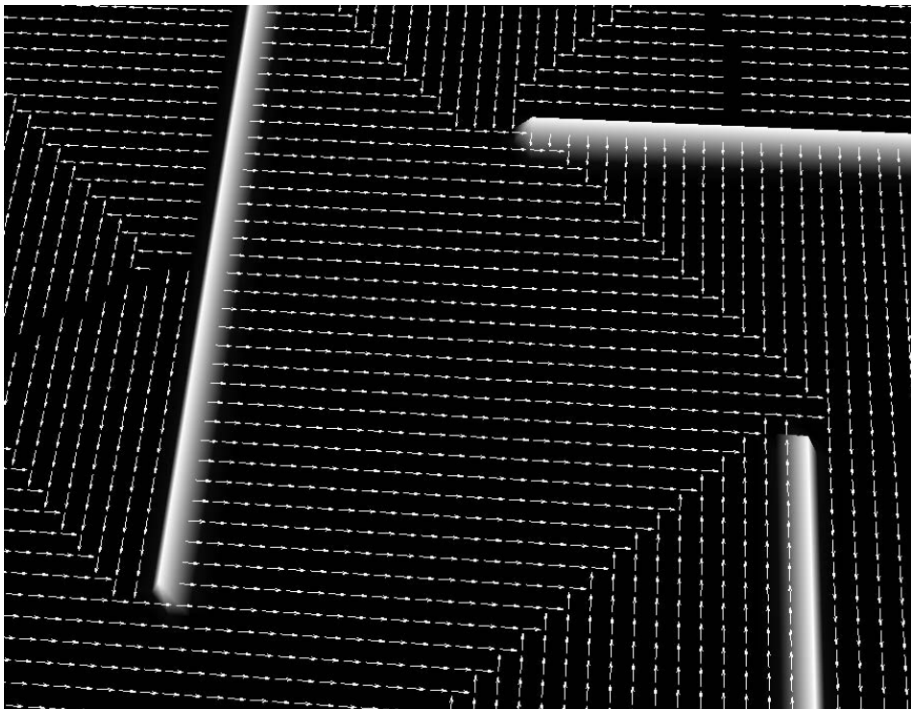
Çalışmalarda, bir kaynak hücreden yayılan yürüyen dalganın çeperinin bu kaynağın konumuna ait bilgiyi ağdaki tüm hücrelere taşıdığı ve bu bilginin kaydedilmesi ile herhangi bir hücreden kaynak hücreye giden en kısa yolun bulunabileceği tespit edilmiştir [8]. Yol probleminin çözümü dalgaların tek bir kaynağı olması, bu kaynak hücrenin matematiksel modelde *u* değişkeni ile oluşturulması önerilmiştir. Böylelikle dalga yayılımı tek bir noktadan başlayıp ağı tamamlayıncaya kadar devam etmektedir. Şekil A.3'de yürüyen dalganın yayılımının farklı iterasyonlardaki durumu gösterilmiştir. Tek bir noktadan başlayan yayılım Şekil A.3-12'de ağın tamamen kaplamasıyla durmuştur. Ağın sunduğu bir avantaj, istenilen hücrelerin durum değerlerinin sabit olarak atanabilmesidir. Böylelikle ağ üzerinde yürüyen dalganın yayılmasına engel olacak engeller oluşturulabilmektedir. Sabit ve dinamik hücrelerin başlangıç değerleri aynı olduğundan bu engeller, dalganın henüz ulaşmadığı kesimlerde fark edilememektedir. Şekil A.4'da bazı hücre değerleri sabitlenmiş ağdaki yürüyen dalganın yayılımı görülmektedir. Engellerin tamamı Şekil A.4-12 de gözükmektedir. [8] numaralı çalışmada verilen yöntem yürüyen dalga çeperlerinin yayılısına dayalıdır. Dalga çeperi sabit hücreler ile ayrılmış koridorlar için kollara ayrılır ve yayılma dinamiği tüm koridorlarda eş zamanlı olarak devam eder.

Dalga çeperinin henüz ulaştığı her hücre için çeperin geliş yönü belirlenir ve kaydedilir. Dalga çeperinin bir hücreye ulaşması, bir hücrenin *x* durumu başlangıç değerinden farklılaşması anlamına gelmektedir. Dalga çeperinin geliş yönü de bu hücrenin genliği en yüksek komşusundan, hücrenin kendisini işaret eden vektördür ve ancak dört ana yönden birisi olabilir. Ağın çalışması esnasında bu yöntem ile

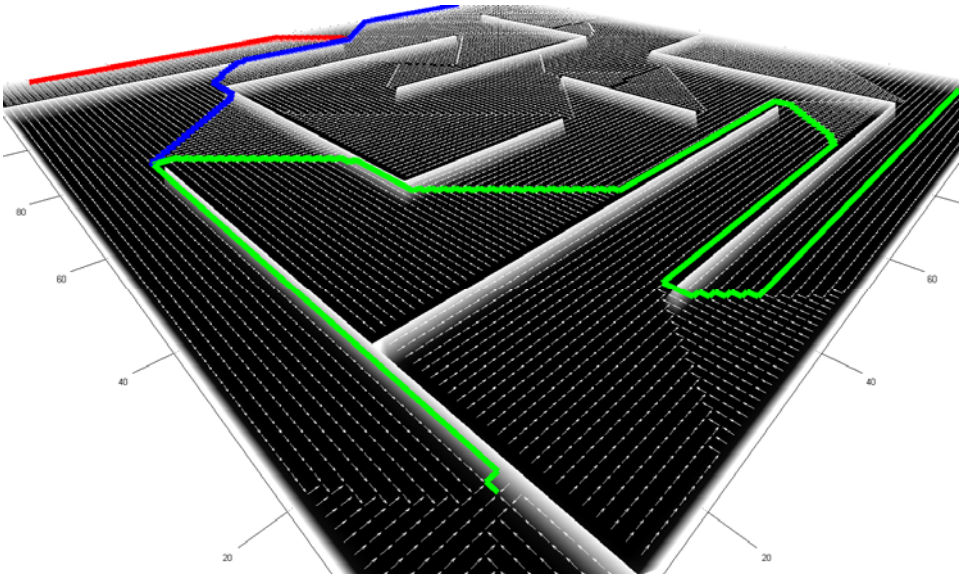
kaydedilen vektörler, en sonunda bir vektör matrisi oluştururlar. Matristeki vektörler kaynaktan hücrelere, dalganın yayılısını işaret ederken; vektörlerin 180^o döndürülmesi sayesinde herhangi bir hücreden dalga kaynağı olan hücreye doğru olan yolu gösterir hale gelirler. Ağda oluşturulan farklı koridorlardan ilerleyen paralel dalga çeperleri koridor sonlarında karsılaşabilirler. Ya da ortada bulunan sabit hücre adasının etrafından dolaşan dalga kolları tekrar kavuşabilirler. Yayılan dalganın özelliği sayesinde bu karsılaşmalarda girişim oluşmaz. Birleşen dalga kolları tek dalga olarak ilerlemeye devam eder. Bir hücreye, kollara ayrılıp tekrar birleşen bir dalga çeperi ulaştığında, hangi kol daha kısa ise elde edilen vektör matrisi o koldan giden bir yolu işaret eder çünkü ayrılan dalga kollarının ilerleme hızları aynı kalır. Ağda dalganın yayılımı tamamlandıktan sonra her hücreden kaynak noktaya giden en kısa yolu veren vektör haritası elde edilir ve bu vektör haritasını kullanarak tasarlanan ağdaki herhangi bir hücreden hedef noktaya ulaşılabilir. Şekil A.4'da tasarlanan ağda yürüyen dalganın yayılımından sonra elde edilen vektör haritası Şekil 3.1'de verilmiştir. Bu haritada kaynak noktaya doğru giden yol ok işaretleri ile gösterilmektedir. Buradan her hücreden kaynak noktaya varılabileceğini göstermektedir. Vektör haritasından yerel olarak bir bölgesinin büyütülmüş hali Şekil 3.2 de verilmiştir. Okların yolu nasıl belirttiği daha net bir şekilde görülmektedir. Şekil 3.3'de vektör haritası üzerindeki rastgele seçilen üç farklı noktadan hedef noktaya giden en kısa yol gösterilmiştir.



Şekil 3.1: Yürüyen dalganın yayılımından sonra oluşan vektör haritası



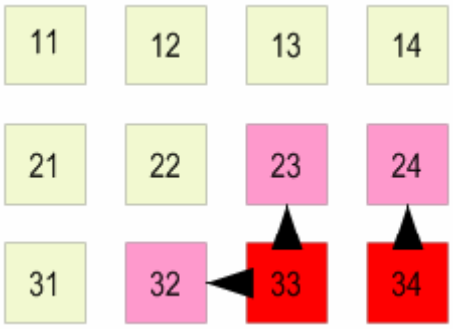
Şekil 3.2: Vektör haritasına yakından bakış



Şekil 3.3: 3 farklı noktadan hedefe giden en kısa yolun gösterimi

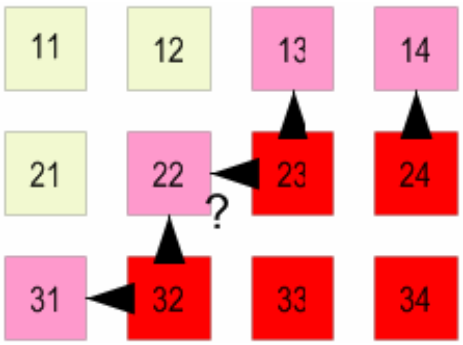
Kırmızı çizgiyle başlayan yol daha sonra mavi çizginin izlediği yola, oradan da yeşil çizginin takip ettiği yola dahil olmaktadır. Algoritma temel olarak ağ üzerinde yayılan dalga çeperinin gradyentinin vektörel olarak hafızalanmasına dayanmaktadır. Başlangıçta durağanlıkta olan HYSA'nın her bir hücresinin üzerine bir dalgaya ait çeper geldiği anda bu dalganın geliş yönü hesaplanabilir. Bu dalga çeperinin o hücrenin bulunduğu yerdeki normal vektörüdür. Bu normal vektörü her hücre için yalnız bir kere hesaplanmalıdır. Çünkü ağ başlangıçta durağanlıkta, dalga yayılmaya

başladıktan sonra da sürekli değişim içindedir. Bizim için problemin çözümünde ise yarayacak dalga çeperi, herhangi bir hücrenin üzerinden geçen ilk dalga çeperidir. İlk dalga çeperinin yakalanması herhangi bir hücre için ilk değer değişiminin algılanması anlamına gelmektedir. Bir hücreye dalga çeperi ulaştığının saptanmasının ardından dalga yönü de o hücrenin dört komşusunun hangisinden dalganın ulaştığının tespiti ile belirlenir. Dört komşuluk ile birbirine bağlı olan hücrelerin üzerine de dalga ancak bu dört yönden birinden ulaşmış olmaktadır. Şekil 3.4'de kırmızı ile gösterilen sağ alt hücrelerden yayılan dalganın pembe ile gösterilen hücrelere hangi vektörler ile ulaştığı gösterilmiştir. Pembe hücreler henüz daha o anda başlangıçtan farklı değer almıştır. Yani dalga çeperi o anda o hücreler üzerindedir.



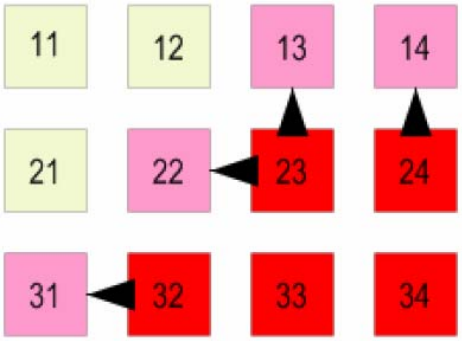
Şekil 3.4: Dalga yayılımı

Dalga, Şekil 3.5'deki gibi yayılmasını sürdürdüğünde aynı yöntem ile vektörler saptanırken 22 etiketli hücrede bir problem ile karşılaşılmıştır. Bu hücreye dalga hangi komşusundan ulaşmaktadır. Bunun tespiti için hemen hangi komşusunun *x* genliğinin daha büyük olduğuna bakılabilir. Bu daha doğru bir yöntem olmakla birlikte, işlem süresini uzatmaktadır. Bu durumda komşulara öncelik atayarak da bir çözüme varılması denenmiştir. Önerilen algoritmada öncelik sırası: sağ komşu > üst komşu > sol komşu > alt komşu şeklindedir.



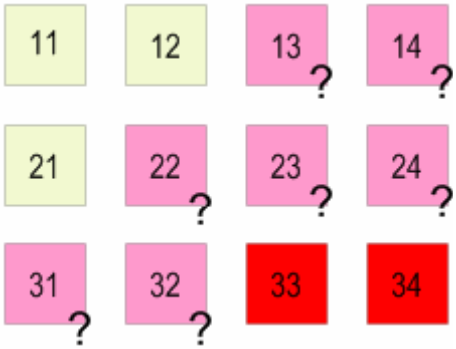
Şekil 3.5: Dört komşuluk problemi

Dolayısıyla Şekil 3.5'deki problemin çözümü Şekil 3.6'daki gibi olmaktadır.



Şekil 3.6: Komşulara öncelik atama

Böylece *u* değişkeni ile belirlenen başlangıç hücresinden dalganın nasıl yayıldığı, dalga çeperinin her ilerleyişinde ilerleme vektörünün saptanması ve bunun 128x128 boyutlu bir vektör matrisinde ilgili hücreye denk düşen yere kaydedilmesi ile olur. Bu algoritmada önemli olan bir nokta dalga çeperinin her bir hücre ilerleyişinde fark edilip vektörün üretilmesi gerektiğidir. Aksi takdirde Şekil 3.7'da gösterilen problem ile yüzleşilir ve yön bulunamaz.

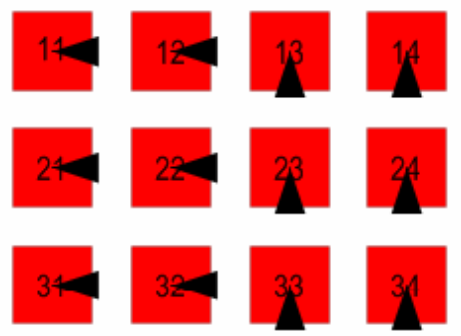


Şekil 3.7: Hızlı dalga ilerleyişi sorunu

Dalganın ilerleyiş hızının en fazla her seferinde bir hücre kadar olacağını garanti edecek sıklıkla HYSA görüntüsü örneklenmelidir. Her iterasyonda gelen ağ görüntüsünü, yani X[k] matrisini, bir değişkende tutulur. Aynı görüntünün bir önceki değeri, yani X[k-1] de bir baksa değişkende tutulmaktadır.

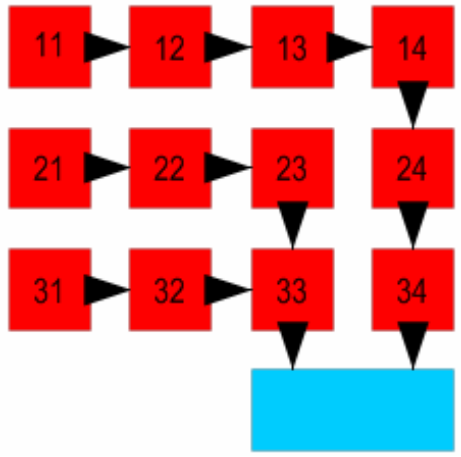
Algoritma koştuktan sonra yani vektörleri barındıran değişkene yeni vektörler işlendikten sonra şimdiki görüntü eski görüntünün üzerine yazılır ve programa bir iterasyon daha koşturması komutu gönderilir.

Bu süreç vektörleri tutan matrisin pasif hücreler hariç tamamıyla dolması ile son bulur. Şekil 3.8 örnek olarak gösterilen ağda son durumu vermektedir.



Şekil 3.8: Son durumdaki vektörlerin durumu

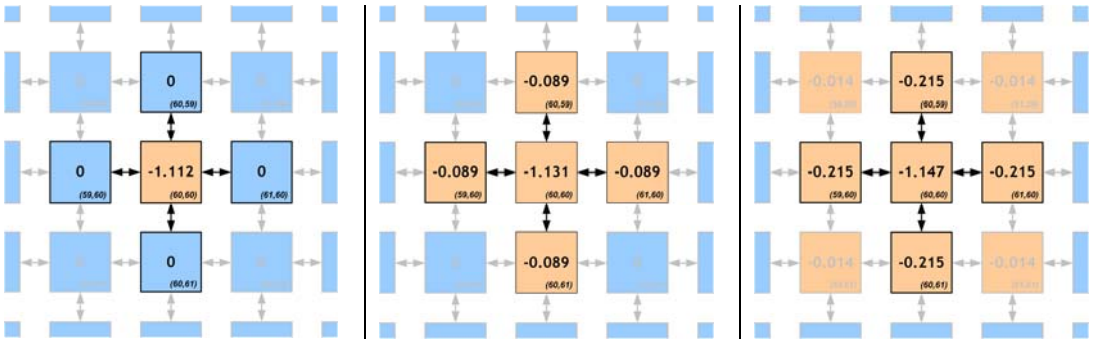
Bu hesaplanan vektörleri tersine döndürerek, herhangi bir hücreden bu döndürülmüş vektörleri takip ederek dalganın başladığı bölgeye ulaşırız. Şekil 3.9 bu örneği göstermektedir.



Şekil 3.9: Başlangıca geri dönme

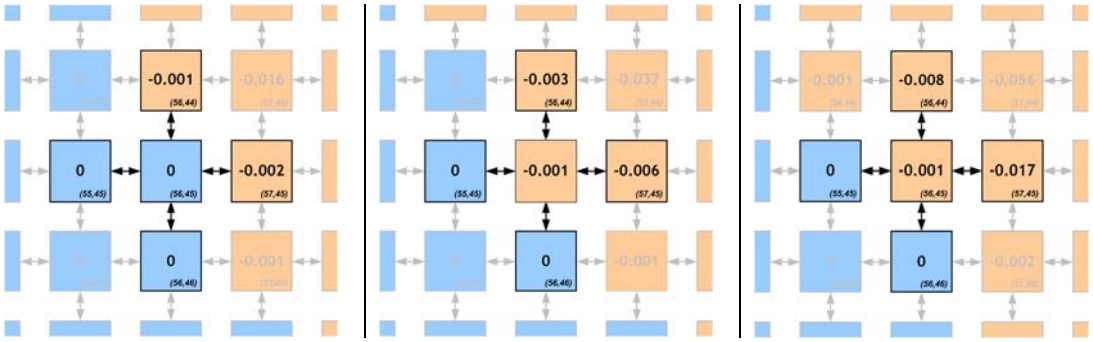
Şekil 3.9'a bakıldığında 11 hücresinden mavi bölgeye ulaşmak için okları takip etmek yeterlidir. Diğer tüm hücreler gibi 23 ya da 31 hücreyi için de bu geçerlidir. Bu algoritma ile elde edilen sonuç, ağ üzerindeki tüm diğer noktalardan başlangıç noktasına olan yolun her bir adımını veren vektörler dizisidir. İşte robot için gerekli olan da tam olarak budur. Robotun gitmesi hedeflenen noktadan yayılan dalgaların ürettiği vektörler ters çevrilir. Robotun bulunduğu noktadan hedeflenen noktaya gitmesi için yapması gereken ilk olarak bulunduğu noktadaki vektörü izleyerek komşu hücreye geçmesidir. Bundan sonraki adımlar ise vardığı hücredeki vektörü izleyerek bir diğer komşuya geçmesidir. Takip işlemi sonunda robot mutlaka dalgaların yayıldığı kaynağa varacaktır. Dalgaları taşıyan hücreler başlangıç değerinden başka bir sonlu değere ulaşmakta ve o değer yaklaşıklığında kalmaktadır. Bu durum, sistemin en temel parçası olan Gevşeme osilatörünün (kare dalga osilatörünün) özel bir davranışıdır. Dalga Çeperi Difüzyonuna Dayalı Algoritma,

taxicab geometrisinde en kısa yolu vermektedir. Taxicab geometrisinde iki nokta arasındaki uzaklık, noktaların koordinatlarını mutlak farkının toplamı olarak tanımlıdır. Öklid uzaklığını yaklaşımı ile en kısa yol sonucunu verememektedir. Bu yüzden algoritmada değişikliğe gidilmiştir Bölümün basından buraya kadar dalga yayılmasında çeperin, bir hücreye transferi esnasında, kaynak hücrenin tespiti yapılırken bir öncelik kullanılmıştır. Algoritmanın bu kısmı değiştirilerek daha iyi bir sonuç araştırılmıştır. Bunun için örnek bi harita X[0] matrisi olarak HYSA'ya yüklenmiştir. Ağın (60,60) hücresi dalga kaynağı olarak seçilmiş ve başlangıç değeri -1.112 olarak ayarlanmıştır. Haritadaki gri bölgelere gelen hücreler pasif hücre olarak etiketlenmiştir. Yürüyen dalga yayan parametreler verilerek iterasyonlara başlanmıştır.Başlangıç hücresinin ve komşularının *x* durumlarının 1., 2. ve 3. iterasyonlarda aldığı değerler Şekil 3.10'de gösterilmektedir.



Şekil 3.10: Kaynak hücre ve komşularının ilk üç iterasyon sonunda x durumları

Dalga ilerleyişi sürdürülmüş ve çeper örnek hücre (56,45)'e vardığında *x* durumlarına bakılmıştır. Şekil 3.11'de bu hücre ve komşuları için *x* durumlarının değerleri verilmiştir. Algoritmadaki küçük değişikliğe göre, örnek hücreye dalgayı ileten hücre, komşularından mutlak x genliği en büyük olan olarak seçilecektir.



Şekil 3.11: (56,45) hücresi ve komşularının 103.- 105. iterasyonlarda x durumları

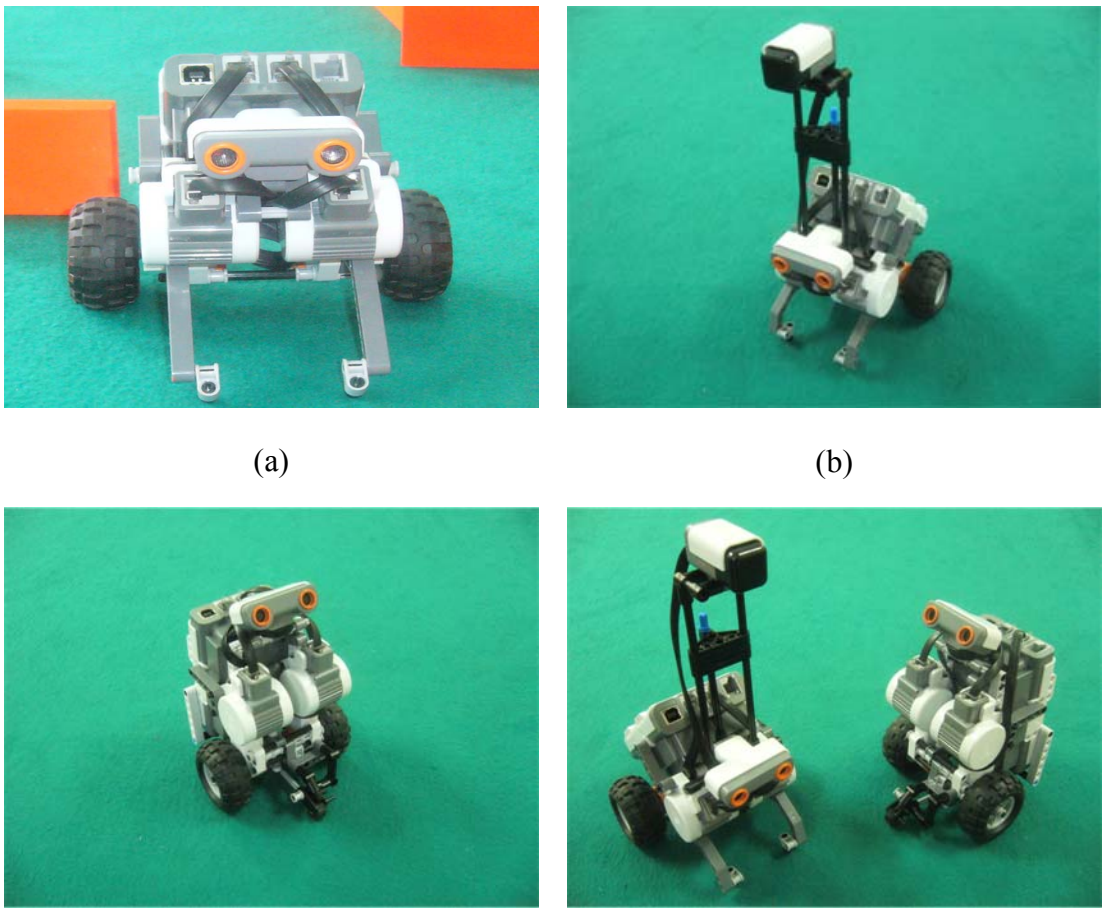
Yani Şekil 3.11'deki durum için (56,45) hücresine yürüyen dalga doğu komşusu olan (57,45)'den ulaşmıştır. Bu yöntemle daha etkin yollar elde edilmiştir.

4. TEST DÜZENEĞİ

Önceki bölümde tasarlanan ağda üretilen dalga çeşitleri ve bu dalgaların nasıl oluşturulduğu ayrıntılı biçimde açıklanmıştır. Bu bölümde yürüyen dalganın robot yönlendirme problemin uygulanması için hazırlanan düzenek anlatılacaktır.

4.1 Robot ve Uygulama Platformu

Uygulamada kullanılmak üzere seçilen robot Lego Mindstorm NXT 2.0 modeli seçilmiştir. Set olarak alınan paketten çıkan lego parçacıklardan uygulama gereksinimlerini karşılayacak şekilde robotlar tasarlanmıştır. Yapılan çalışmanın ihtiyacına göre yapılan robot tasarımları Şekil 4.1'de verilmiştir.



(a) (b)

(c) (d)

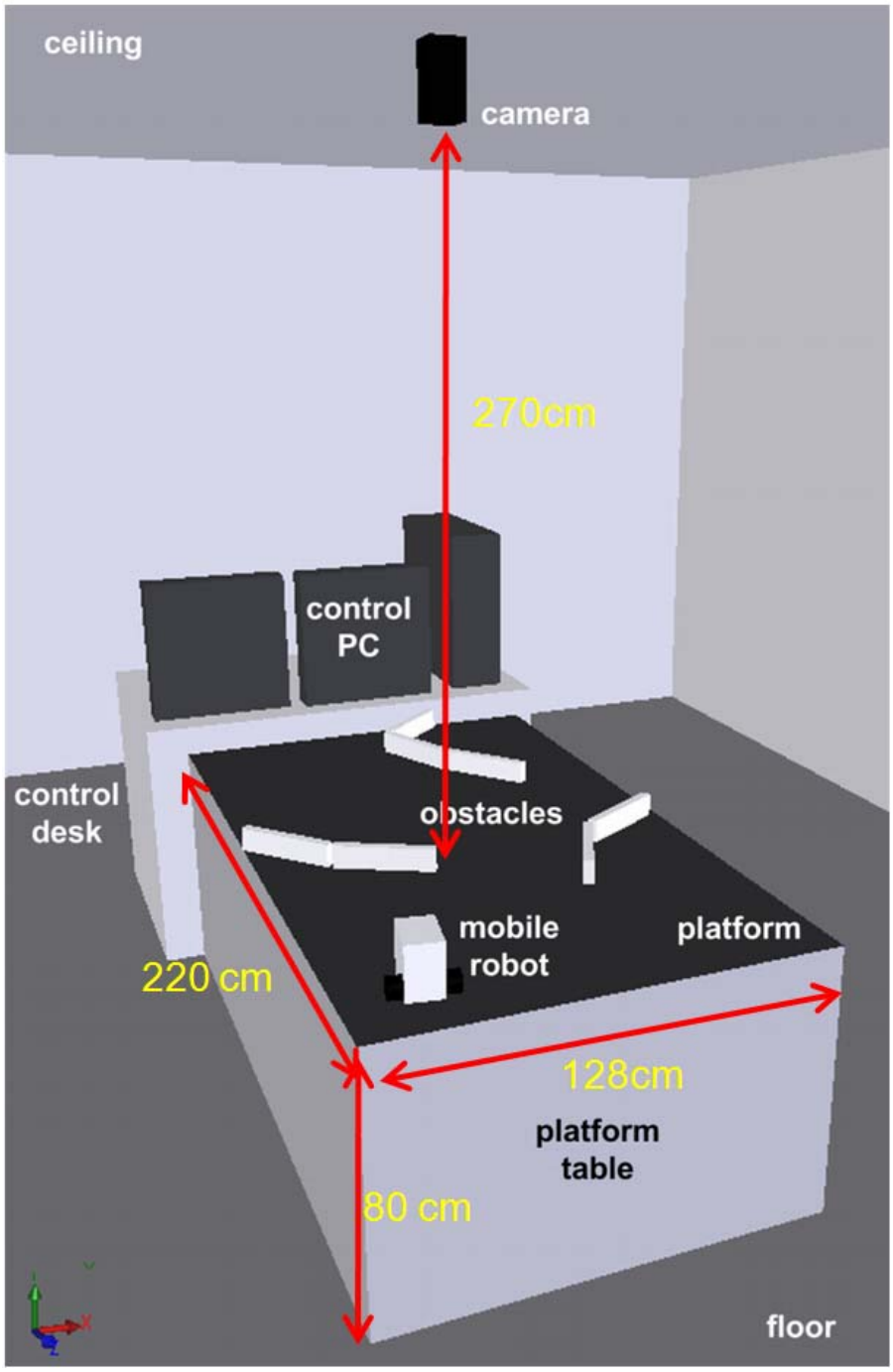
Şekil 4.1: Tasarlan robotlar (a-d)

Çalışmanın ilk aşamasında bir robotun önceden belirlenmiş sabit bir noktaya gitmesi planlanmıştır. Bunun için Şekil 4.1-a deki tasarım yapılmıştır. İleriki aşamalarda kaçan robotu takip eden robot gerçekleşmesi için robot sayısının ikiye çıkarılması gerekmiş ve Şekil 4.1-b ve Şekil 4.1-c deki tasarımlar yapılmıştır. Takipçi robota yön bilgisinin daha hızlı elde edilmesi için Şekil 4.1-b de görüldüğü üzere manyetik pusula takılmıştır. Kaçan robot Şekil 4.1-c de görüldüğü gibi daha dik bir iskelet yapısı ile tasarlanmıştır. Bunun nedeni görüntü işleme sırasında iki robotu birbirinden platform üzerinde kapladığı alana göre ayırt edebilmektir.

Çalışmada robot olarak Lego Mindstorm NXT 2.0 seçilmesinin en önemli nedeni Bluetooth üzerinden haberleşmenin sağlanabilmesidir. Robot ile bilgisayar arasındaki veri alışverişinin Bluetooth üzerinden kablosuz bir şekilde sağlanması kablolardan kaynaklanabilecek robotun hareketini kısıtlayabilmesi muhtemel sorunları ortadan kaldırmıştır. Robotla Bluetooth üzerinden haberleşmek için yapılması gerekenler altbölüm 4.1.1'de ayrıntılarıyla anlatılmıştır.

Sistemin üzerinde uygulandığı Şekil 4.2'de görülen platformun boyutları 220 x 128 cm ve yerden 80 cm yüksekliktedir. Robota engel oluşturmak için çeşitli boyutlarda ahşap engeller kullanılmıştır. Görüntü işlemede kolaylık olması için engel oluşturmada kullanılacak tüm parçalar turuncu renge boyanmıştır. Platform üzerindeki engellerin sayısı ve boyutu daha fazla parça kullanılarak arttırılabilir. Zemin üzerinde istenilen şekil oluşturulabilir. Çalışma zeminin özellikle mat olması istenmiş ve yeşil bir ton seçilmiştir. Bunun temel sebebi üzerine düşen ışığı yansıtmasının istenmemesidir. Mat zemin görüntü işlemede avantaj sağlamaktadır. Işığın etkisi az olduğunda yansımalar az olmakta bu sayede işlemeler sonucunda net sonuçlar alınmaktadır. Kurulan platformun görüntüsü tavana yerleştirilen kamerayla bilgisayara aktarılmaktadır. Kamera platformun tamamını görebilecek şekilde tavana monte edilmiştir. Platformdan 270 cm yüksekliktedir ve bilgisayara kablo ile bağlanmıştır. Bu, verinin kameradan bilgisayara daha hızlı bir şekilde iletilmesini sağlamakta, dolayısıyla görüntünün alınmasında zaman kazandırmaktadır. Kameranın konumlandırılmasında ışığın etkisi de göz önünde bulundurulmuştur. Platform tavandaki aydınlatmanın tam altına yerleştirilmiştir. Çalışma anında aydınlatma açık tutulmaktadır. Böylelikle camdan gelen gün ışığının etkisi azalmaktadır. Sistemi daha hızlı gözlemlemek ve çalışmalarda hız kazanmak amacıyla kontrol bilgisayarına iki seri monitör bağlanmıştır.

Merkez bilgisayarın masaüstü bu sayede genişletilmiştir. Bu genişletmeye, algoritma geliştirme esnasında hem platformun durumunu izlemek hem de kodların düzenlenmesini sağlamak için ihtiyaç duyulmuştur. Görsel çalışma alanın artması çalışanların olayları hızlı yorumlamasını ve karar vermesini sağlayarak performans artışı sağlamıştır. Çalışmalar sırasında kullanılan dosyaların çokluğu göz önünde bulundurulduğunda klasörler arası geçişlerde ve uygulamaların görüntülenmesinde çift monitör uygulaması zaman kazandırmaktadır.

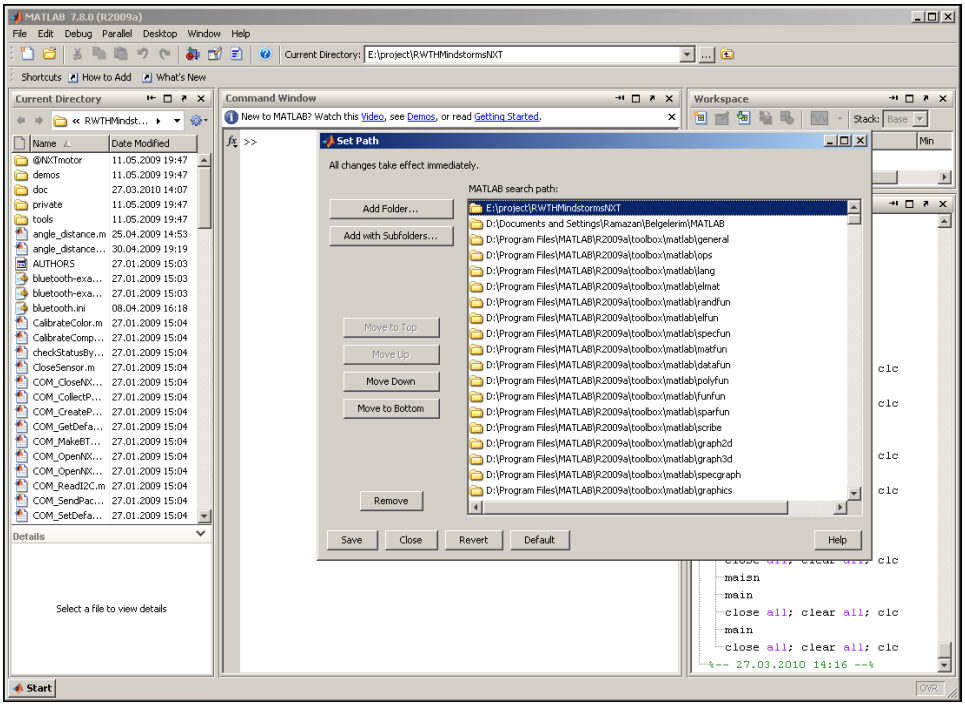


Şekil 4.2: Uygulama Platformu

4.1.1 Ana bilgisayarla robot arasındaki bağlantının kurulması

Bu bölümde robotla Bluetooth üzerinden haberleşmek için yapılması gerekenler sırasıyla anlatılacaktır. Ana bilgisayardan robotu Matlab üzerinden yönlendirmek için Mindstormun Matlab ile uyumlu olan http://www.mindstorms.rwth-aachen.de/trac/wiki/Download adresinden indirilen ücretsiz RWTHMindstormsNXT toolbox 4.00 sürümü kullanılmıştır. Bu toolbox ile Matlabdan Bluetooth kullanarak haberleşmek mümkün olmaktadır. HYSA'nın Matlab da tasarlanmış olmasının yanında robotu da Matlab üzerinden yönlendirmek projenin başarısı açısından önemlidir. Tüm sistemin tek bir program altında çalışması tasarlanan proje modullerinin birbiriyle daha hızlı veri alışverişi yapmasını sağlamıştır. Özellikle yürüyen dalganın yayılımından sonra elde edilen vektör haritasından robotun rotasını çıkarılması ile robota ilgili komutları gönderilmesi aşamasında iki modulün aynı program altından çalışması veri alışverişi açısından sistemi hızlandırmıştır.

Matlabda tüm sistemi çalıştırabilmek için indirilen toolbox ile diğer modüllerin kodları aynı klasörde olmalıdır. Bu çok fazla dosyanın bir arada olmasına neden olacağından dosyaları istenildiği zaman arayıp bulmak zaman alacaktır. Bunun yerine Matlabın bir özelliğini kullanarak bu sorundan kurtulabilir. Matlabın menüsünden file> setpath seçilir ve RwthmindstormNXT klasörünün olduğu adres Şekil 4.3 de gösterildiği buraya eklenir.



Şekil 4.3: Matlabda klasöre yol atama

Sistem çalıştırıldığı zaman, Matlab bu adresteki toolboxıda kontrol edecektir. Böylelikle diğer modülleri ait kodlar ile toolboxın kodları bir arada olmak zorunda kalmayacaktır.

Matlabda bu pratik uygulama yapıldıktan sonra robotlarla Bluetooth haberleşmesinin açılması gerekmektedir. Bu çalışmada IVT BlueSoleil in 6.4.249.0 sürümü kullanılmıştır ve ara yüzü Şekil 4.4 de gösterildiği gibidir. BlueSoleil çalıştırıldığında Şekil 4.4-a daki ekran gelecektir. Ortadaki turuncu daireye tıklandığında çevredeki Bluetooth cihazlarını aramaya başlar. Bu arada robotlar açık ve Bluetoothları aktif olmalıdır. Robotlar bulununca ara yüzde Şekil 4.4-b deki gibi görünür. Ara yüzdeki her bir robot simgesinin üzerine sağ tıklanarak "pair device" komutu seçilir. Bilgisayarla robotu eşleştirmeye yarayan bu komut, robotun ekranındaki şifrenin Şekil 4.4-c de gösterildiği gibi girilmesiyle robotların bilgisayar tarafından tanınmasını sağlar. Şekil 4.4-d görüldüğü gibi her bir robot simgesinin sağ alt tarafında zincir işareti çıkmalıdır. Daha sonra her bir robot simgesi Şekil 4.4-e gösterildiği gibi sağ tıklanarak "connect bluetooth serial port(#)" seçilmelidir. Bilgisayardan robota Bluetooth üzerinden bağlantı sağlanmış olacaktır. Şekil 4.4-f nin gözükmesi bilgisayarla robotun haberleştiğini göstermektedir.

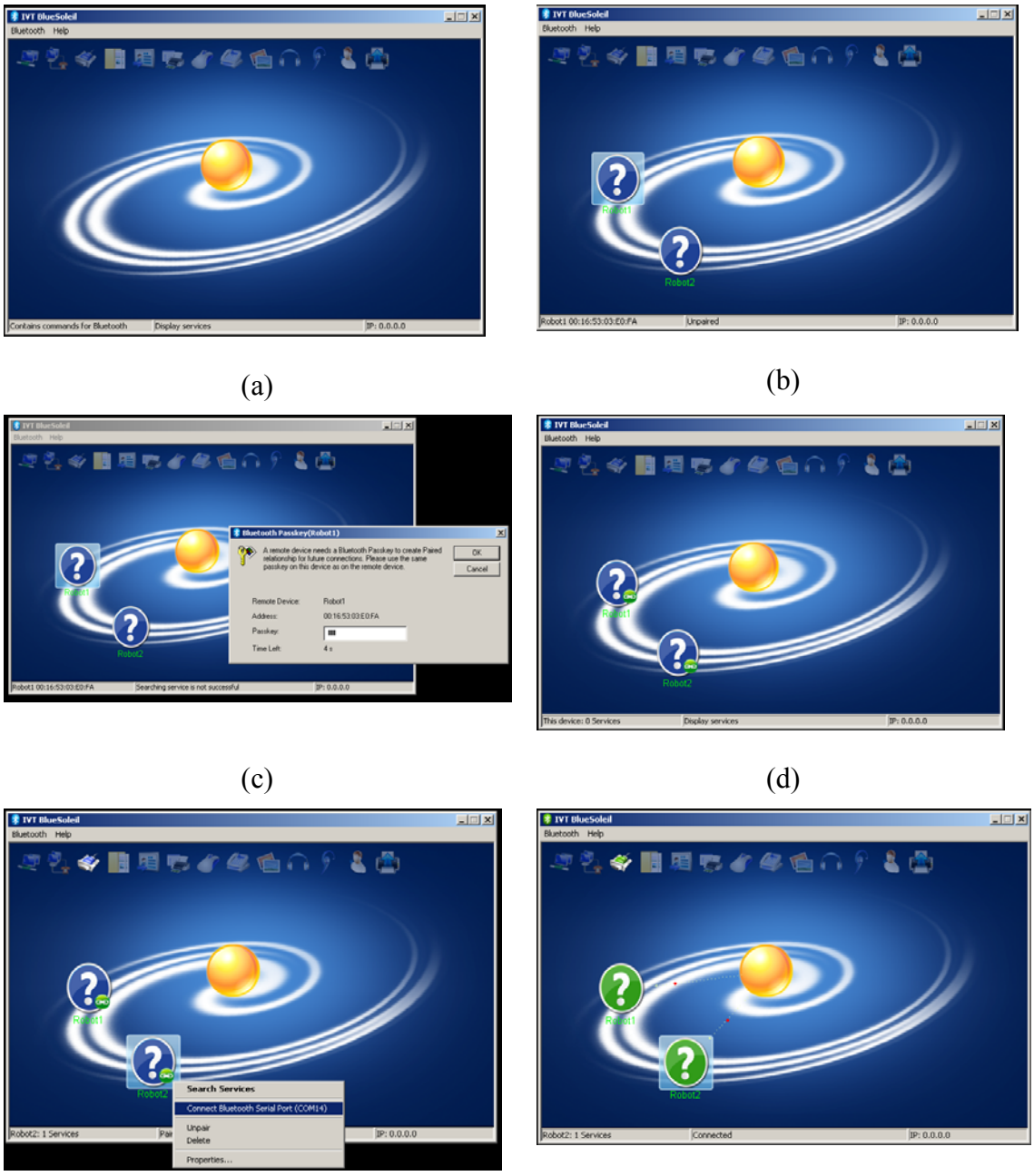
Bilgisayarla robot arasında haberleşme sağlandıktan sonra, bu haberleşme kanalını Matlab ile de ilişkilendirmek gerekmektedir. Matlabdan komut göndermek için öncelikle komut satırına

```
> handle = COM_OpenNXT('bluetooth.ini', 'check');
```

yazılması gerekmektedir. Buradaki 'bluetooth.ini' her robot için yazılmış olması gereken dosyalardır ve içerisinde şu bilgiler yer almaktadır.

```
[Bluetooth]
SerialPort=COM"#"
BaudRate=9600
DataBits=8
SendSendPause=5
SendReceivePause=30
Timeout=2
```

Elimizde 2 robot olduğundan ve farklı comlar üzerinden çalışacağından robot1.ini ve robot2.ini dosyaları oluşturulur ve sadece SerialPort=COM# satırıldaki '#' yerine Şekil 4.4-f de robot simgelere sağ tıklanıp 'status' den robotların 'com' değerleri girilir. Buradaki robot1.ini ve robot2.ini isimleri keyfi adlandırmadır. İstenildiği gibi değiştirilebilir.



(a) (b)

(c) (d)

(e) (f)

Şekil 4.4: Bluetooth arayüzünün kullanılması

Elimizde iki robot olduğundan Matlabın komut satırına

```
> handle1 = COM_OpenNXT('robot1.ini', 'check');
```

```
> handle2 = COM_OpenNXT('robot2.ini', 'check');
```

Komutları girilerek robotlar için Matlabın ‘com’ larla ilişkilenmesi beklenir. Burada bu komutların Matlab tarafından yerine getirilmesi 5-10 saniye sürmektedir. Bu sadece ilk başta yapılacak daha sonra bu komutlar tekrar girilmeyeceği için bu zaman kaybı tekrar yaşanmayacaktır.

```
> COM_SetDefaultNXT(handle);
```

komutuyla çalışılacak handle yani açılan COM aktif oluyor. Elimizde iki robot olduğundan

```
> COM_SetDefaultNXT(handle1);
> COM_SetDefaultNXT(handle2);
```

komutları ile kullanılacak olan robot belirlenir . Bu çok hızlı olduğundan COM açıp kapamayla zaman kaybedilmeyecektir. Yukarıdaki işlemler yapıldığında robotla haberleşme sağlanabilecektir.

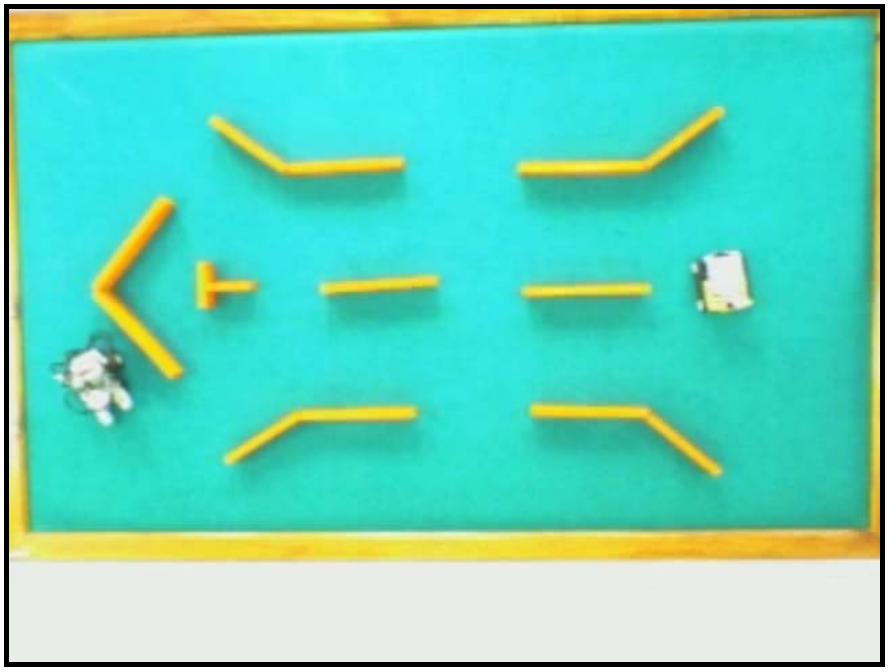
4.2 Görüntü işleme

Robot için aranan en kısa yolu bulan, HYSA ile oluşturulan aktif dalgalardır. Robotun dolaşabileceği uzay olan platform düzlemi görüntüsü, aktif dalgaların yayılacağı ağ üzerine başlangıçta yüklenir. Fakat bu kameradan alınan görüntü ile ağa yüklenen arasında fark bulunmaktadır. Ağa yüklenen görüntü aslında hücrelerin başlangıç koşullarını ifade eden bir matristir. Geliştirilen ve kullanılan yöntemde ağdaki hücrelere üç farklı başlangıç koşulu yüklenmektedir. Bu farklı başlangıç koşulları ile ağ üzerinde başlangıçta üç tip hücre oluşturulmaktadır. Bunlar:

- Aktif hücreler,
- Pasif (ölü) hücreler,
- Kaynak (hedef) hücredir.

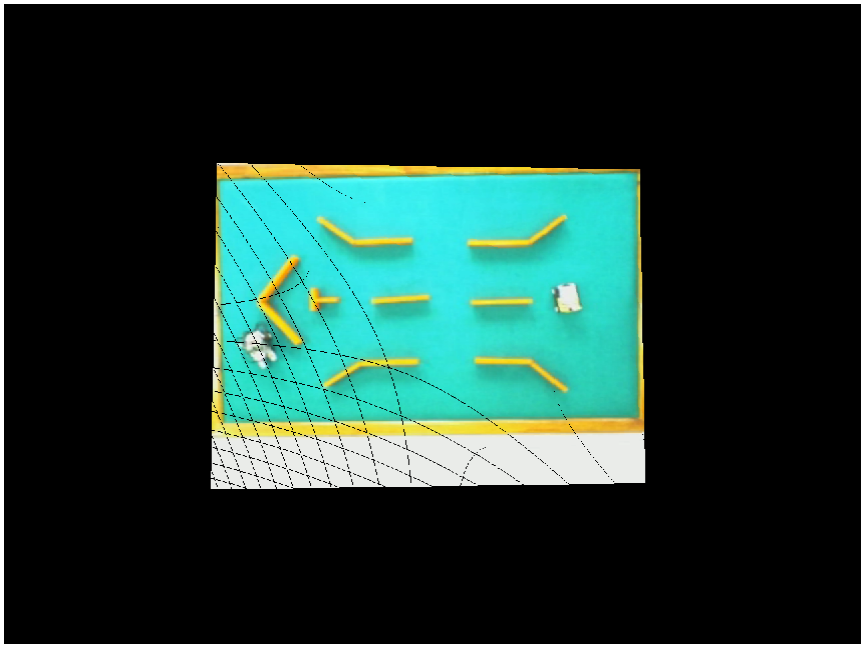
Aktif hücreler platform görüntüsünde robotun üzerinde bulunabileceği, geçebileceği, engel bulundurmayan piksellere karşılık gelir. Pasif ya da ölü hücreler ise robotun geçiş yapamayacağı üzerinde engeller bulunan piksellere karşılık gelir. Tek robotla yapılan çalışmalar sırasında kaynak hücre tepe kamera görüntüsünden elde edilmeden kullanıcı tarafından manüel olarak belirlenirdi. Çalışmaların ilerletilip iki robotlu kaçma kovalama senaryosu uygulanır hale geldiğinde kaynak hücre de tepe kameradan alınan görüntüden elde edilir olmuştur. Tezin çalışma safhası boyunca uygulanan görüntü işleme yöntemlerinde de değişikliğe gidilmiştir. Bu bölümde kullanılan görüntü işleme metotları ve yapılan değişiklikler nedenleriyle açıklanmıştır. Platform açıklanırken ifade edildiği gibi engeller turuncu renkte, zemin ise yeşil renktedir. Tepe kameranın çektiği görüntü robotu ve engeller ile platform düzlemi arasındaki ara renkli bölgeleri, engellerin gölgelerini de içerir. Ağa

yüklenecek başlangıç koşulu görüntüsü ise sadece aktif hücreleri oluşturan siyah, pasif hücreleri oluşturan yeşil ve kaynak hücreyi oluşturan üçüncü bir renkten oluşmaktadır. Kaynak hücrenin ağ başlangıç görüntüsündeki rengi, kaynak hücrenin başlangıçta alacağı renk ile ilişkilidir. Kameradan alınan görüntüden HYSA'ya yüklenebilecek uygun görüntüye geçiş için yapılan işlemeler ön görüntü işlemeleri olarak adlandırılmıştır. Aşağıda ön görüntü işleme basamakları, açıklayıcı figürleri ile birlikte anlatılmaktadır.Görüntü işlemede ilk yaklaşım kameradan alınacak en büyük çözünürlüklü görüntüyü almak ve işlemeyi bu görüntü üzerinden devam ettirmektedir. Alınan görüntünün kalitesi işlemeden sonra çıkacak görüntünün kalitesini yükseltecektir. Tavana monte edilmiş kameradan alınabilecek en yüksek çözünürlüklü görüntü 640x480 boyutlarında Şekil 4.5'de gözükmektedir



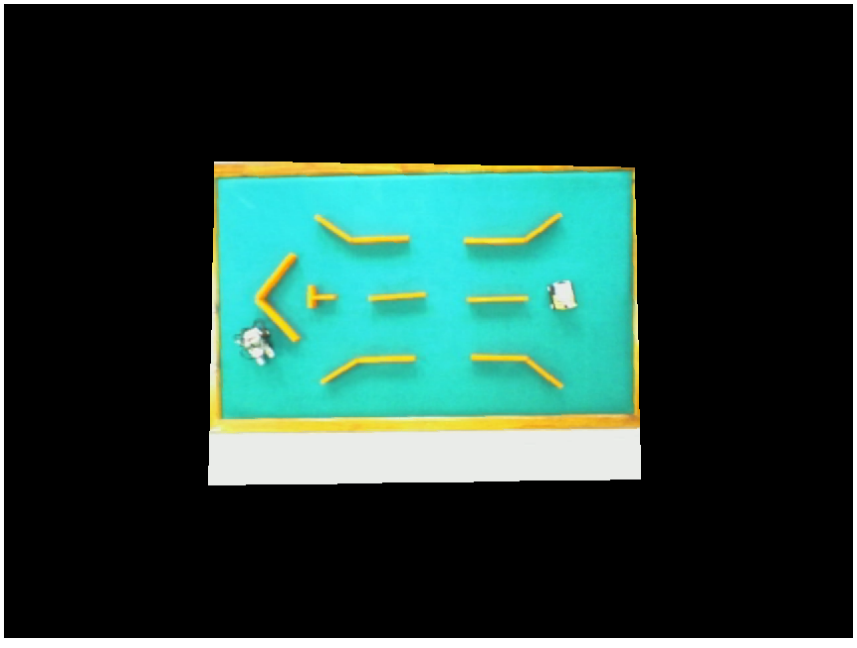
Şekil 4.5: 640x480 boyutlarındaki en büyük çözünürlüklü görüntü

Görüntüde iki robot gözükmektedir. Engeller turuncu renktedir ve alt taraftaki beyaz bölüm ise gereksiz alandır. Görüntü işleme sırasında buralar atılacaktır. Burada dikkat edilmesi gereken nokta alınan görüntünün her üç eksende de eğik olmasıdır. Kameranın tavana eğik monte edilmesi ve platformun yere düz bir şekilde oturtulmamış olması eğikliğin temel sebebidir. Dolayısıyla bu eğikliğin görüntü işleme sırasında giderilmesi gerekmektedir. Bunun için alınan görüntü kendisinden daha büyük bir zemine oturtulup projektif dönüşüm uygulanır. Yapılan işlem sonucunda elde edilen görüntü Şekil 4.6'da verilmiştir. Projektif dönüşüm esnasında görüntü üzerinde bazı noktaların değeri hesaplanamaz.



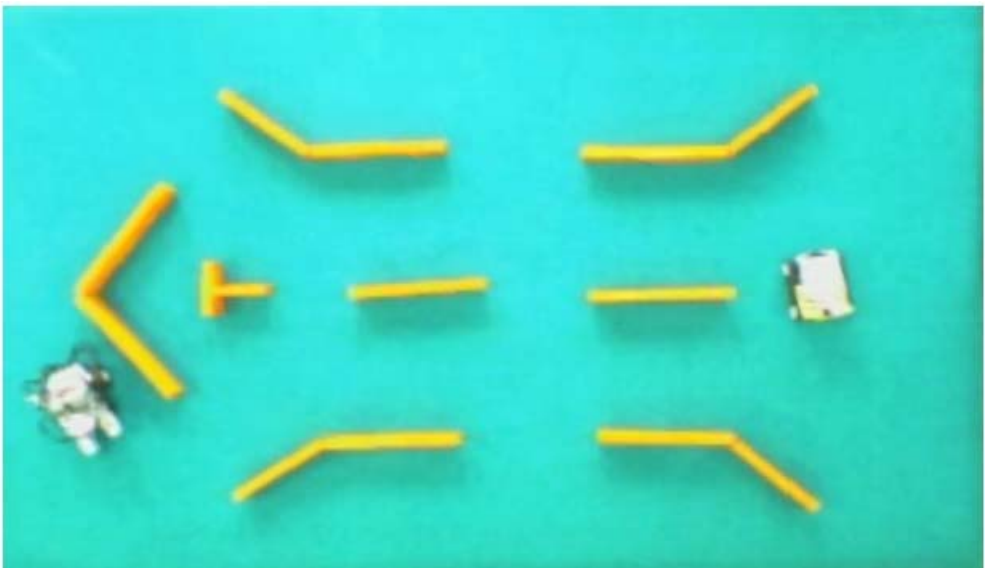
Şekil 4.6: Asıl görüntünün büyük bir düzlem içerisinde eğikliğinin giderilmesi

Şekil 4.6'da görüntü üzerindeki siyah yaylar değeri hesaplanamayan piksellerden oluşmaktadır. Bu değerler, interpolasyon ile kestirilir ve tepe görüntüsü Şekil 4.7'deki hali alır.



Şekil 4.7: İnterpolasyon ile siyah çizgilerin yok edilmesi

Alınan görüntüdeki eğiklik giderildikten sonra görüntü işlenmeye hazır hale gelmiştir. Şekil 4.7'ten istenilen alan çıkartılır. Sonuç Şekil 4.8'deki gibidir. Robotları ve engelleri ayırt etmek için kullanılan yöntem yuvarlaklık metriğinden faydalanmaktır. Metrik (4.1)'de gösterildiği gibi tanımlanmaktadır.

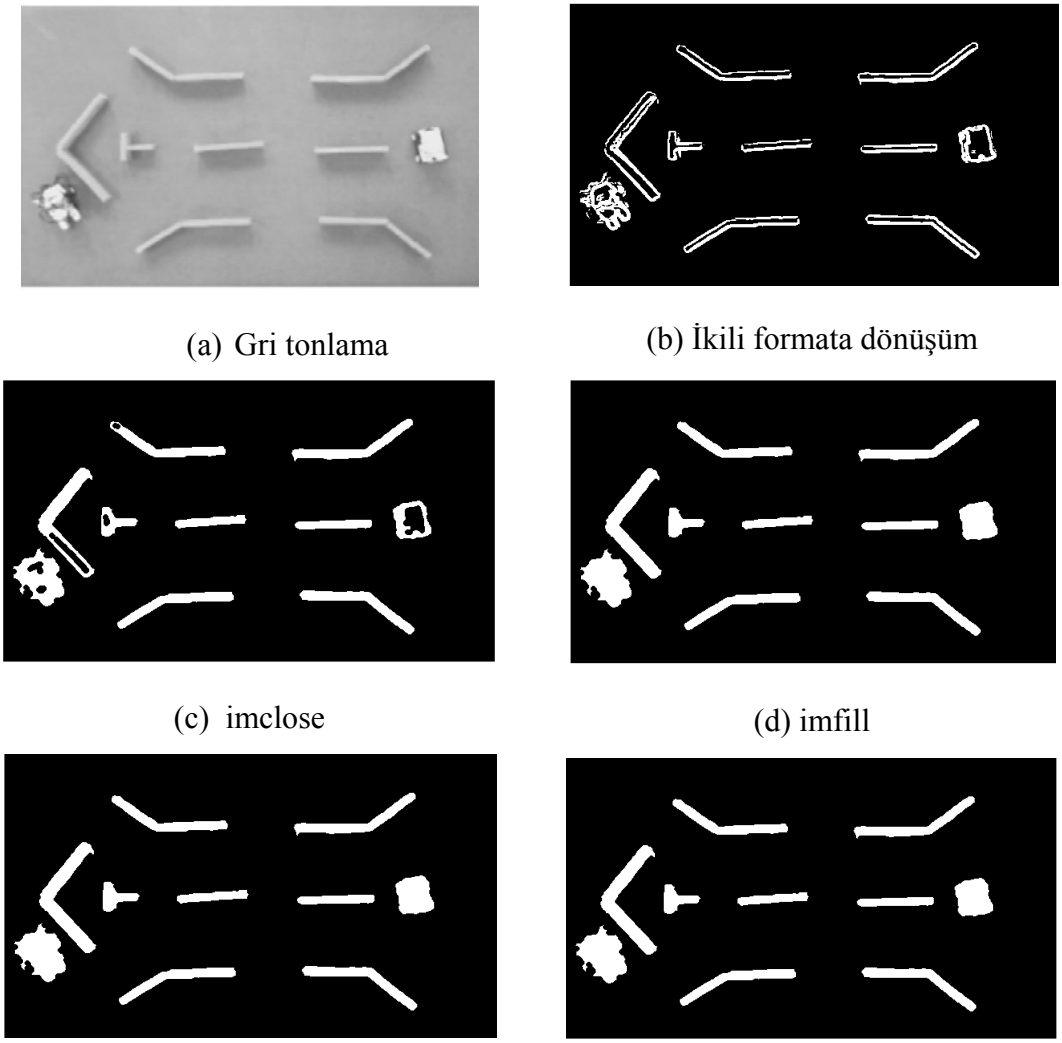


Şekil 4.8: Eğikliği giderilmiş ve kenarları kırpılmış işlenmeye hazır görüntü

$$metrik = \frac{4x\pi xalan}{çevre^2} \qquad \textbf{(4.1)}$$

Bir cismin metriğinin 1 olması onun tam yuvarlak olduğunu gösterir. Şekil 4.8'e bakıldığında engeller robota göre daha dikdörtgen, robotlar ise daha bir kareseldir. Bu robotların metriğinin engellere göre daha yüksek çıkacağını gösterir. Metrik hesabından gidilerek robotlar ve engeller birbirinden rahatlıkla ayırt edilebilir. Bunu için resmin gri tonlamaya çevrilmesi gerekmektedir. Metriğin hesaplanması için siyah beyaz görüntüdeki tüm gürültüler kaldırıldıktan sonra şekillerin etiketlenmesi gerekmekte ardından da etiketlenen şekillerin resim üzerinde kapladığı alanları ve sahip olduğu çevre uzunlukları çıkartılarak metrikler hesaplanmalıdır. Daha sonra belli bir kesim değerine göre engeller robotlardan ayrılabilir. Bunun yapmak için gereken işlemler Şekil 4.9'da gösterilmiştir. Şekil 4.9–a da gri tonlamaya çevrilmiş görüntü resmi gözükmektedir. Ardından Şekil 4.9–b de görüldüğü üzere resim binary formata çevrilir. Şekillerin iç kısımlarında boşluk olduğu görülmektedir. Bunlar "*imclose"* komutuyla şekillerin çizgileri arasındaki boşluklar kapatılır. Resmin geldiği durum Şekil 4.9-c de gösterilmiştir. Şekillerin içi *"imfill"* komutuyla beyaza boyanır.

Şekil 4.9-d gösterilen şeklin durumu alan hesaplaması için önemli bir durumdur. Bu aşamaya geldiğinde resimde oluşabilecek gürültüler göz önünde bulundurulmalıdır. *"bwareaopen"* komutuyla resminde olabilecek gürültüler giderilir. Şekil 4.9-e şeklin getirilmesi gereken önemli bir yerdir.



(a) Gri tonlama (b) İkili formata dönüşüm

(c) imclose (d) imfill

(e) bwareaopen (f) label

Şekil 4.9: Görüntü üzerinde yapılan işlemler

Bundan sonraki aşamalarda resim üzerindeki şekillerin etiketlenmesi, sınır ve alan bilgilerinin elde edilmesi gerekmektedir. Şekil 4.9-f de resim etiketlenmiştir. Şekil 4.10'da her bir şeklin metriği hesaplanmış ve şekil üzerinde gösterilmiştir. Şekil incelenirse engellerin metriği 0.4 ün altındadır. Bu değer kullanarak metriği 0.4 ün altında ve üstünde olan şekiller birbirinden ayrılabilir.

Şekil 4.10-b de robotlar görüntüden çıkartılmış, Şekil 4.10-c de görüntünün boyutları 128x75 e düşürülmüş ve Şekil 4.10-d kameradan alınan görüntü en son aşamasına getirilmiş ve ağa yüklenecek şekilde 128x128 boyutuna getirilmiştir.

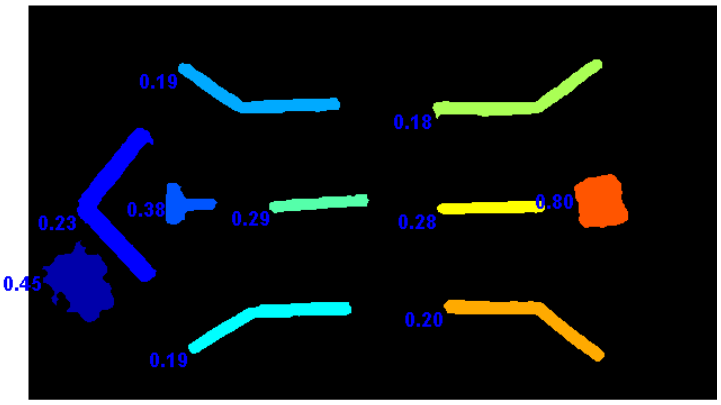
Yukarıda anlatılan yöntemle, görüntü işlemede uygulanan ilk metottur. Bu yöntemle robot ve engel resimleri istenilen kalitede elde edilmektedir fakat bu yöntemin getirdiği çekince bahsedilen tüm bu işlemlerin istenilenden fazla zaman almasıdır.

Geçen süre ortalama 10.33 saniyedir. Bu dinamik ortamda yapılacak bir çalışma için görüntü işlemeye bu kadar süre ayrılması mümkün değildir. Bunun üzerine yeni görüntü işleme yöntemleri geliştirilmeye gidilmiştir.

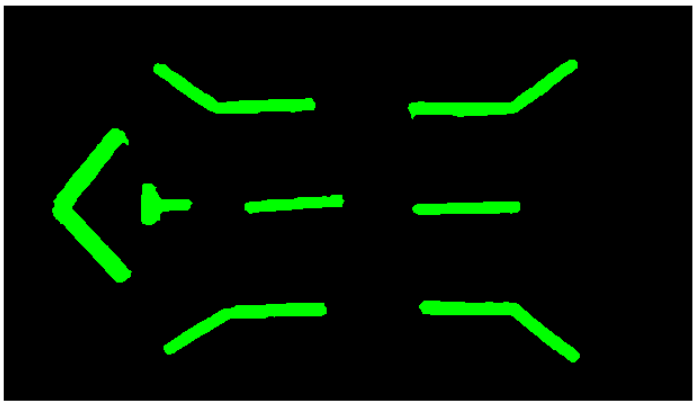
İlk olarak kameradan alınan görüntünün boyutu düşürüldü. 640x480 yerine 160x120 boyutlu görüntüyle çalışılmaya başlanmıştır. Yüksek çözünürlüklü görüntünün yüksek hesaplama gücü gerektiren görüntü işleme teknikleri ile daha doğru bilgi edinilmesini sağlayacağı bilinmektedir. Fakat bu çalışmada, alınan görüntü nihayetinde 128 x 128 boyutlu HYSA başlangıç görüntüsüne dönüştürüleceği için düşük çözünürlüklü görüntü ile başlanmış ve yeterli başarımda sonuçlar üretilmiştir. Bu yüzden 160 x 120 boyutlu ve renkli çekilen tepe kamera görüntüsü yeterlidir.Şekil 4.11'de tepe kamera ile çekilen görüntülerden biri örnek olarak verilmiştir.

160 x 120 boyutlu görüntünün nasıl pikselize göründüğüne dikkat edilmelidir. Bu görüntüde engel duvarlar 2 ya da 3 piksel genişliğinde görünmektedir. Bu genişliğe engel ile platform zemini arasındaki geçiş pikselini de eklersek 4 ya da 5 piksele çıktığını söyleyebiliriz. Geçiş pikselleri ile sarı renkli engel piksellerini çevreleyen daha koyu renkli turuncu ya da kahverengi pikseller ifade edilmektedir. Robotun serbestçe dolaşabildiği platform zemini (yeşil bölge) incelendiğinde tüm resim için gayet homojen olduğu söylenebilir. Kimi yerlerde bir miktar engel gölgesi ile koyulaşmalar olsa da bunlar ileride açıklanacağı şekilde işlendiğinde önemsiz kalmaktadır. Homojen zemin rengi, platformun tepeden aydınlatılması ile sağlanmıştır. Tepe aydınlatmanın zemin üzerinde yansımaması da zeminin mat özelliği sayesinde olmaktadır. Aksi takdirde çoğu uygulamada karşılaşılan aydınlatma yansımaları problemi ile bu çalışmada da karşılaşılmaması mümkün değildi. Ön görüntü işlemlerinde ilk yapılan işleme görüntü boyutunun daha da küçültülmesi olmaktadır. Görüntünün uzun kenarı yani sütun sayısı 160 pikselden 128 piksele düşürülmektedir. 4:3 görüntü oranını korumak için de kısa kenar 120 pikselden 96 piksele düşürülmektedir. Bu küçültme işlemi için *irmesize* fonksiyonu kullanılır. Bu ve bundan sonra kullanılacak olan tüm fonksiyonlar, merkez bilgisayar üzerinde çalışmakta olan MATLAB ortamında geçerli olan fonksiyonlardır.

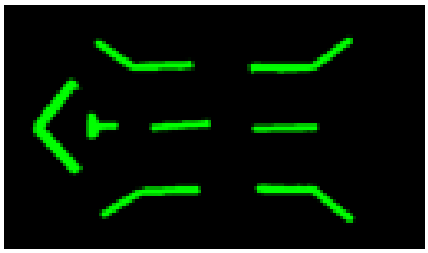
Şekil 4.12'deki görüntü küçültülmüş boyutlu görüntüdür. Şekil 4.11'deki görüntüye göre daha fazla pikselize olmuştur. Fakat hala daha engeller zeminden ayırt edilebilecek kadar kalın ve farklı renktedir.



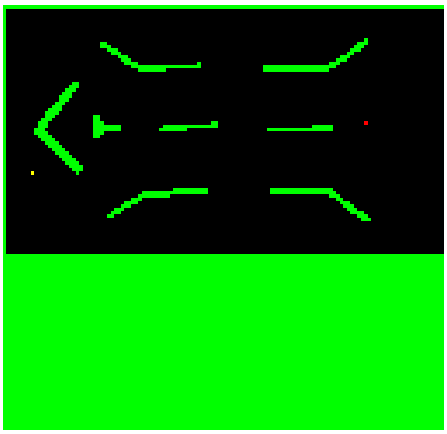
(a)



(b)

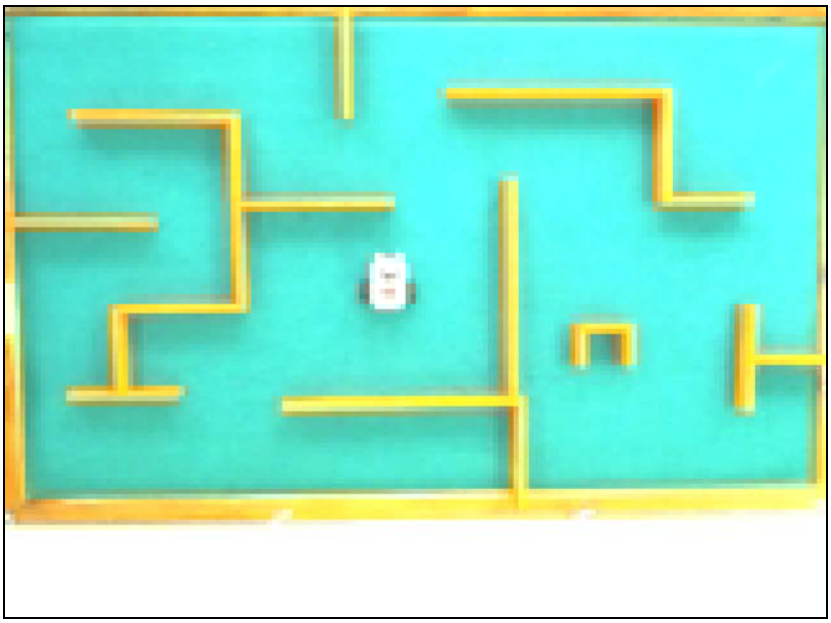


(c)

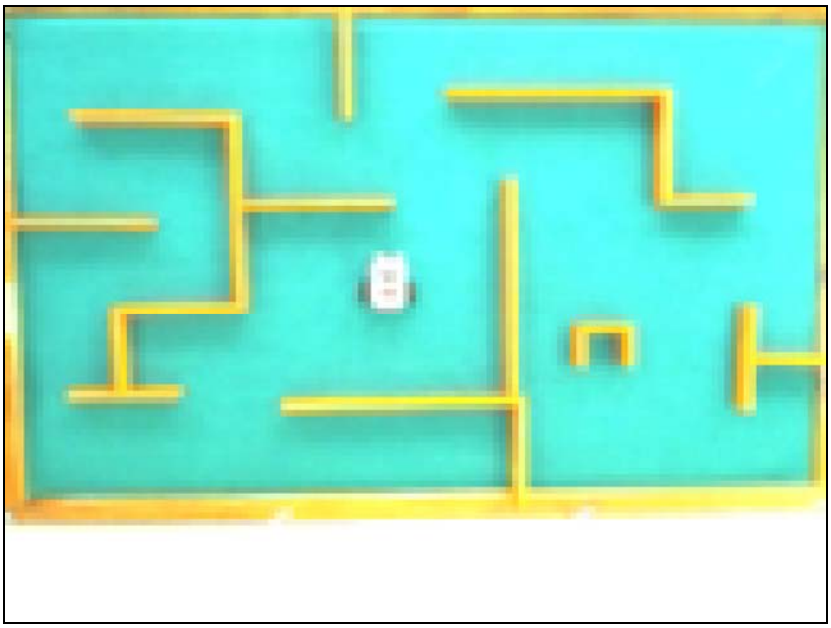


(d)

Şekil 4.10: Alınan görüntünün ağa yüklenmeye hazır hale gelmesi



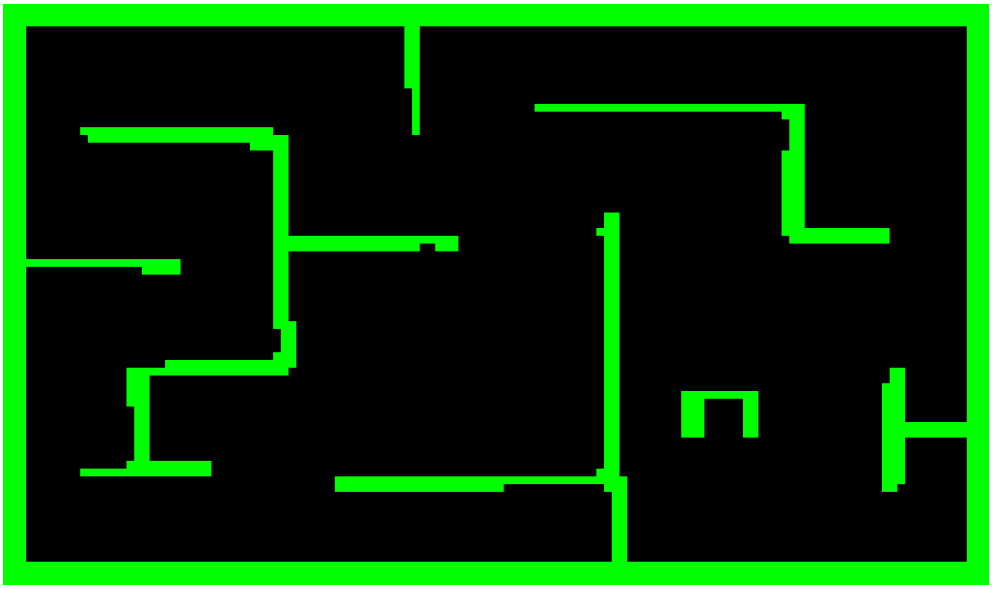
Şekil 4.11: Tepe kamerdan alınan görüntü 160x120



Şekil 4.12: 128x96 boyutuna küçültülmüş görüntü

Şekil 4.12'deki küçültülmüş görüntünün altındaki beyaz bölge, platformun yerleştirildiği laboratuarın zeminidir. Tepe aydınlatması ve kameranın beyaz ve renk dengesini platformun orta yerine göre yapması sebebiyle laboratuar zemini tamamen parlamıştır. Sıradaki adım görüntüden alttaki bu beyaz bandın atılmasıdır. Bu işlem engellerin tespit edilmesi ile birlikte yapılmaktadır. Tepe kamera, aslında turuncu olan engelleri sarıya yakın göstermektedir. Platform üzerindeki engeller, robot ve zemin renk olarak birbirinden yine de ayrıktır. Bu yöntemde bir önceki yöntemden farklı olarak görüntünün düzeltmesi yapılmamıştır. Eğik olan görüntüyü düzeltilmesi

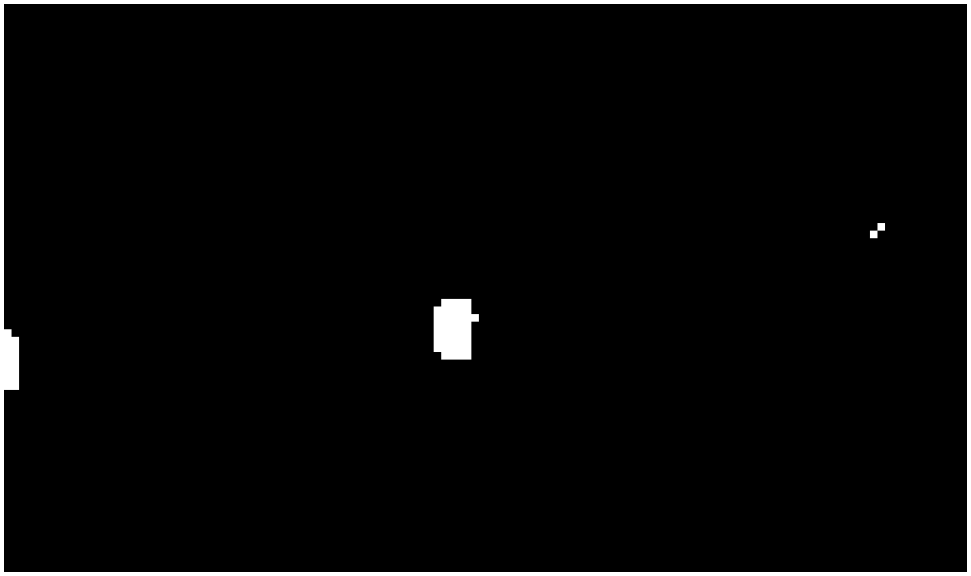
ve sonrasında da oluşan siyah çizgileri yok edilmesi görüntü işlemede istenilenden fazla zaman almaktadır. Yapılan ölçümlerde resim döndürme 1.27 saniye, döndürme sonrasında oluşan siyah çizgileri yok etmek ise 4.24 saniye aldığı görülmüştür. Görüntü işlemede ilk uygulanan yöntem piksellerin sahip olduğu R,G ve B bileşenlerine göre sınıflandırma yapmaktır. Zemine ait piksellerde G ve B bileşenleri yüksek değerde iken R bileşeni oldukça düşüktür. Engellere ait piksellerde de R ve G bileşenleri yüksek iken B bileşeni düşüktür. Robotta ise her üç bileşende oldukça yüksek değerlere sahiptir. Şekil 4.12'deki görüntüden önce engeller tespit edilmiştir. Engellerin tespiti için kırpılacak beyaz banda kadar tüm pikseller taranmış ve R>200, G>150, B<170 koşulu aranmıştır. Bu koşula uyan pikseller yeşile (RGB=0,255,0), uymayanlar siyaha boyanmıştır. Ardından elde edilen görüntü sanal bir engel ile çerçevelenmiş yani, dörtkenarı 3 piksel kalınlığında yeşile boyanmıştır. Engel tespiti işleminin sonucunda Şekil 4.13'deki görüntüye ulaşılmıştır.



Şekil 4.13: Kırpılmış engel görüntüsü

Bu engel görüntüsünde engellerin tamamının tespit edildiğine ancak gerçek kalınlıklarından daha az kalınlıkta tespit edildiğine dikkat edilmelidir. Şekil 4.12'deki görüntüden engellerin tespit edilmesi gibi robotun da tespit edilmesi gerçekleştirilmiştir. Robotun tespit edilmesi işlemi aynen engellerin tespiti gibi, alttaki kırpılacak beyaz banda kadar koşula uyan piksellerin beyaza, uymayanların siyaha boyanması ile yapılmıştır. Robota ait pikselin koşulu her piksel için R>190, G>190, B>190 şeklindedir. Bu işlem ile elde edilen robotun tespit edildiği görüntü Şekil 4.14'de verilmiştir. Bu görüntüde ortada robotun olduğu büyük bir beyaz

bölge, kenarlarda ise robot olma renk koşullarını sağlayan ama aslında robot olmayan hatalı küçük bölgeler görülmektedir. Bu hatalı bölgeler platform çerçevesinde ve engeller üzerinde parlama ile oluşan bölgelerdir. Bunların giderilmesi takip eden işlemler içerisinde gerçekleştirilecektir.



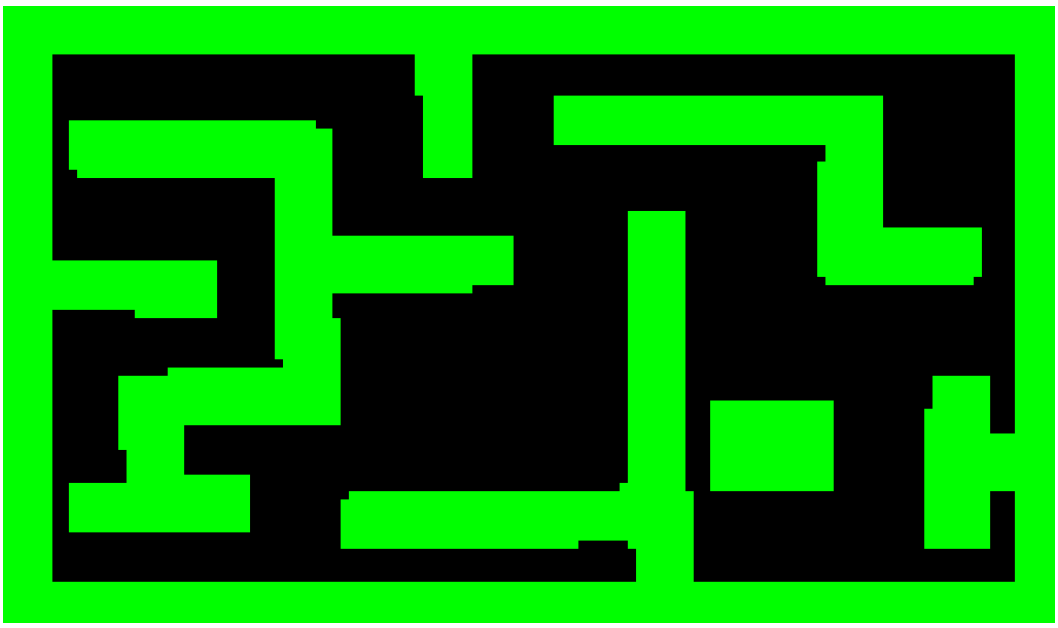
Şekil 4.14: Kırpılmış robot görüntüsü

Şekil 4.12'deki görüntüden kırpılıp alınan bölgenin boyutu 128 x 75 pikseldir. Alttaki atılan beyaz bölge ise 128 x 21 pikseldir. Bu noktadan sonra Şekil 4.13'deki engel görüntüsü morfolojik genişletme işlemine tabi tutulmuştur ve Şekil 4.15'deki sonuç elde edilmiştir. Genişletmenin temel sebebi robotun büyük bir nesne yerine bir nokta ile ifade edilmesi halinde, engellere çarpmasını önlemektir. Robot ifadesi robotun en dış piksellerinden orta noktasına doğru küçülürken, engel ifadesi de bu küçülme miktarında platform zeminine doğru büyümektedir.

Sonuçta elde edilen şişirilmiş engeller arasından bulunan yollar, robotun gerçekten gidebileceği yollardır. Bu yöntem engel geçitlerinden robotun geçebilmesini de garanti etmektedir ve benzer uygulamalarda sıkça kullanılan bir yaklaşımdır. Robot görüntüsündeki gürültülerin giderilmesi için de ilk olarak Şekil 4.14'deki görüntüye morfolojik açma işlemi uygulanmıştır. Açma işleminde kullanılan çekirdek 3 x 3 piksel içerisine sığan bir artı şeklidir. Morfolojik açma işleminde, görüntü içindeki nesnelerden açma operatörünü içine sığdıramayanlar temizlenir.

Bu açma işlemi ile robot görüntüsündeki küçük tek ya da birkaç piksellik gürültülerin giderilmesi sağlanmıştır. Elde edilen görüntü Şekil 4.16'de verilmiştir.

Ardından Şekil 4.15 daki genişletilmiş engellerin tersi ile Şekil 4.16'deki gürültülü robot görüntüsü lojik VE işlemine tabi tutulur.

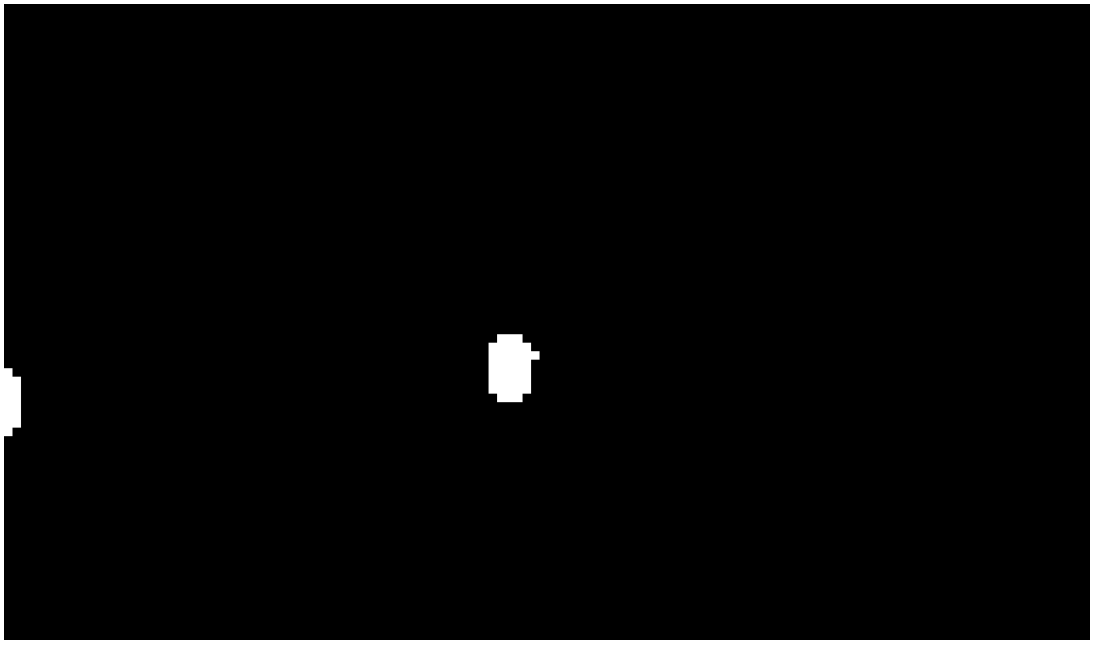


Şekil 4.15: Genişletilmiş engeller görüntüsü

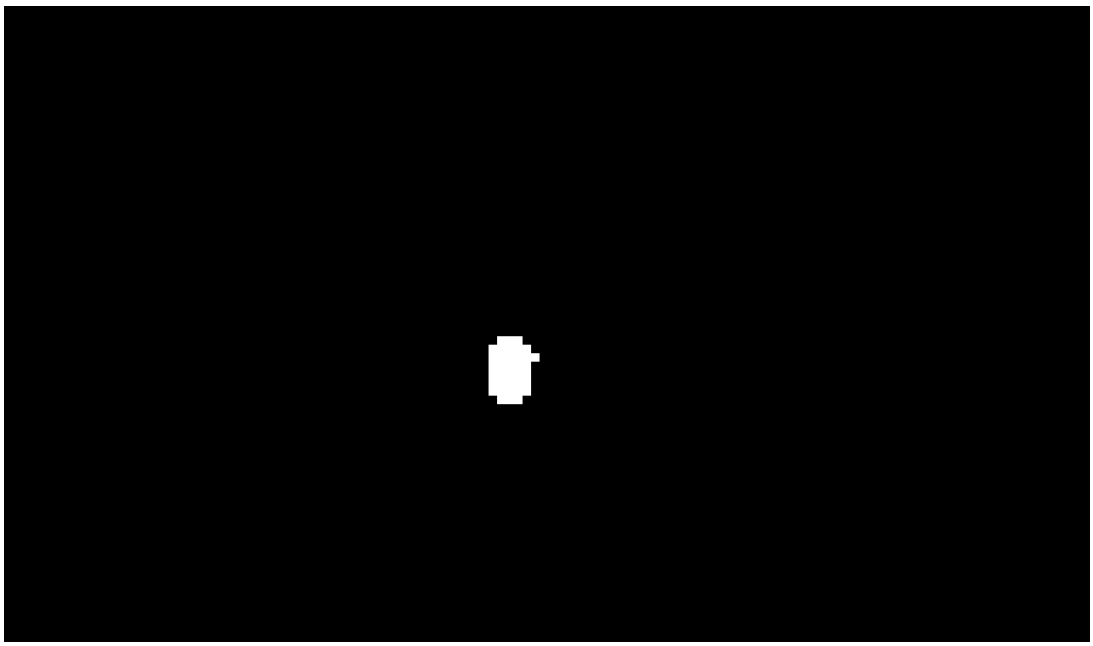
Bu işlem ile robot görüntüsünde, engellere denk gelen gürültülü bölgeler de temizlenmiş olur. Bu adımın sonuç görüntüsü Şekil 4.17'de verilmiştir. Şekil 4.15'de sunulan genişletilmiş engeller görüntüsünün alt kısmına 53 siyah renkli satır daha eklenerek boyutu 128 x 128'e getirilir. Bu boyut HYSA için kabul edilebilen boyuttur. Bu genişletme işleminin ardından görüntü üzerinde dalganın yayılacağı piksel değeri HYSA için -1.122 değerine karşılık gelen renge boyanır. Elde edilen görüntü HYSA'ya yollanmaya hazırdır. Bu yöntemle görüntü işlemeden istenilen sonuçlar alınmıştır. Robot ve engel görüntüsü % 100 net olmasa da bir önceki metoda göre daha iyi bir performans sergilemiştir. Engellerin görüntüsü genişletildiği için engeller elde edilmesinde oluşacak hata tolera edilebilmektedir. Robotların ise sadece konumu istendiği için konumların gerçek yerinden ± 5 piksel yanlış çıkması ne dalga yayılımında ne de robotun hedefe yönlendirilmesinde büyük sorunlar ortaya çıkarmayacaktır.

Bir önceki yönteme oranla işlemler daha kısa sürmekte yaklaşık süre bir saniyenin altındadır. Bu şekilde çalışmalar belli bir süre sürdürüldü. Robotun sahip olduğu yön bilgisi görüntü işlemeyle elde edilmeye çalışıldığından sahip olunan yöntemin yeterli olmadığı görünmüştür. Robotun sahip olduğu yön açısının net bir şekilde çıkarılması robotun doğru bir şekilde yönlendirilmesi için hayati önem taşımaktadır. Kullanılan

yöntemle robotların konumu doğru çıkartılmasına rağmen sahip oldukları şekil tam olarak çıkarılamamaktadır. Takipçi robotun yön açısını da görüntü işlemeyle elde edebilmek için farklı bir uygulamaya gidilmiştir ve son yöntem olan görüntü bölütleme yöntemine başvurulmuştur.



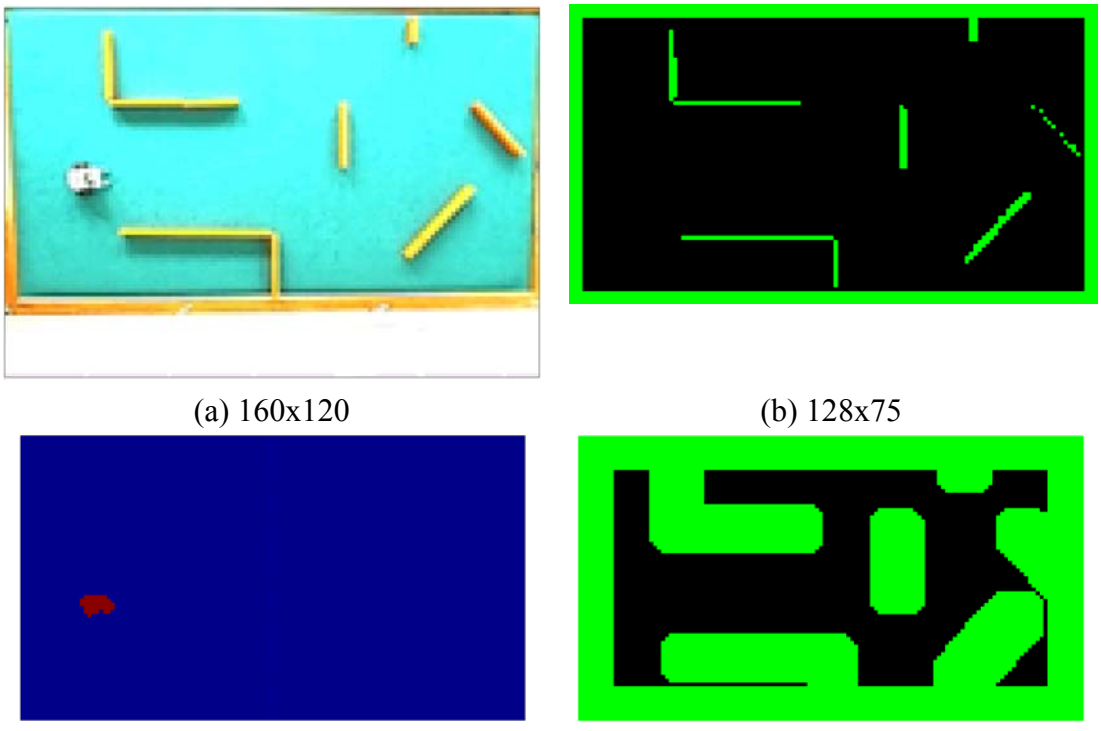
Şekil 4.16: Morfolojik açma işlemi uygulanmış robot görüntüsü



Şekil 4.17: Gürültüden arındırılmış robot görüntüsü

İlk bölütleme adımında robotlar ve engeller platform zemininden ayrılmış, ikinci bölütleme adımında, ilk adımdaki sonuç kullanılarak robotlar ve engeller birbirinden ayrılmıştır. Hedef robot takipçi robota göre daha küçük tasarlanmıştır. Bu sayede iki

robotu da barındıran resimde, robotlar birbirinden izdüşüm alanları ile ayrılmaktadır. Her iki robotun izdüşümlerinin alan merkezleri bulunmakta ve takipçi ya da hedef olarak etiketlenmektedir. Engellere ait resim elde edildikten sonra morfolojik işlemlerle resimdeki engeller genişletilmektedir. Bu, noktasal ifade edilen fakat belirli bir hacme sahip olan takipçi robotun engellere çapmadan hareket etmesini sağlar. Engeller ile otonom hareket eden takipçi robot arasında güvenli mesafenin korunması gerekmektedir. Genişletilmiş engel görüntüsü hareketli hedef takibi döngüsü içerisinde yürüyen dalga ağına sabitlenmiş hücreleri belirtmesi için yüklenir. Dalga kaynağı olacak hücre de hedef robotun merkezine denk düşmektedir. Burada dikkat edilecek husus, üretilen engel görüntüsü çözünürlüğü ile yürüyen dalga ağı boyutunun örtüşmesi ve hedef robotu temsil eden noktanın bu çözünürlükteki resme göre belirlenmesi gerektiğidir. Tavan kamerasından platformun kuşbakışı görüntüsü en fazla 640x480 çözünürlükte alınabilmektedir. Görüntü çözünürlüğü, daha islemeye başlamadan 128x96 boyutuna düşürülmekte, ardından sadece platform görüntüsü kalacak şekilde 128x75 boyutlarına kırpılmaktadır. Bu çalışmada yürüyen dalga ağının boyutu da 128x75 olarak ayarlanmıştır. Düşük çözünürlüklü çalışmanın sebebi işlem sürelerini kısaltmak, güncelleme sıklığını dolayısıyla sistemin gerçek zamanlı performansını arttırmaktır.



(a) 160x120 (b) 128x75

(c) 128x75 (d) 128x75

Şekil 4.18: Görüntü bölütleme ile yöntemiyle robot ve engel resimlerinin çıkarımı

4.3 Robot Hareketinin Belirlenmesi

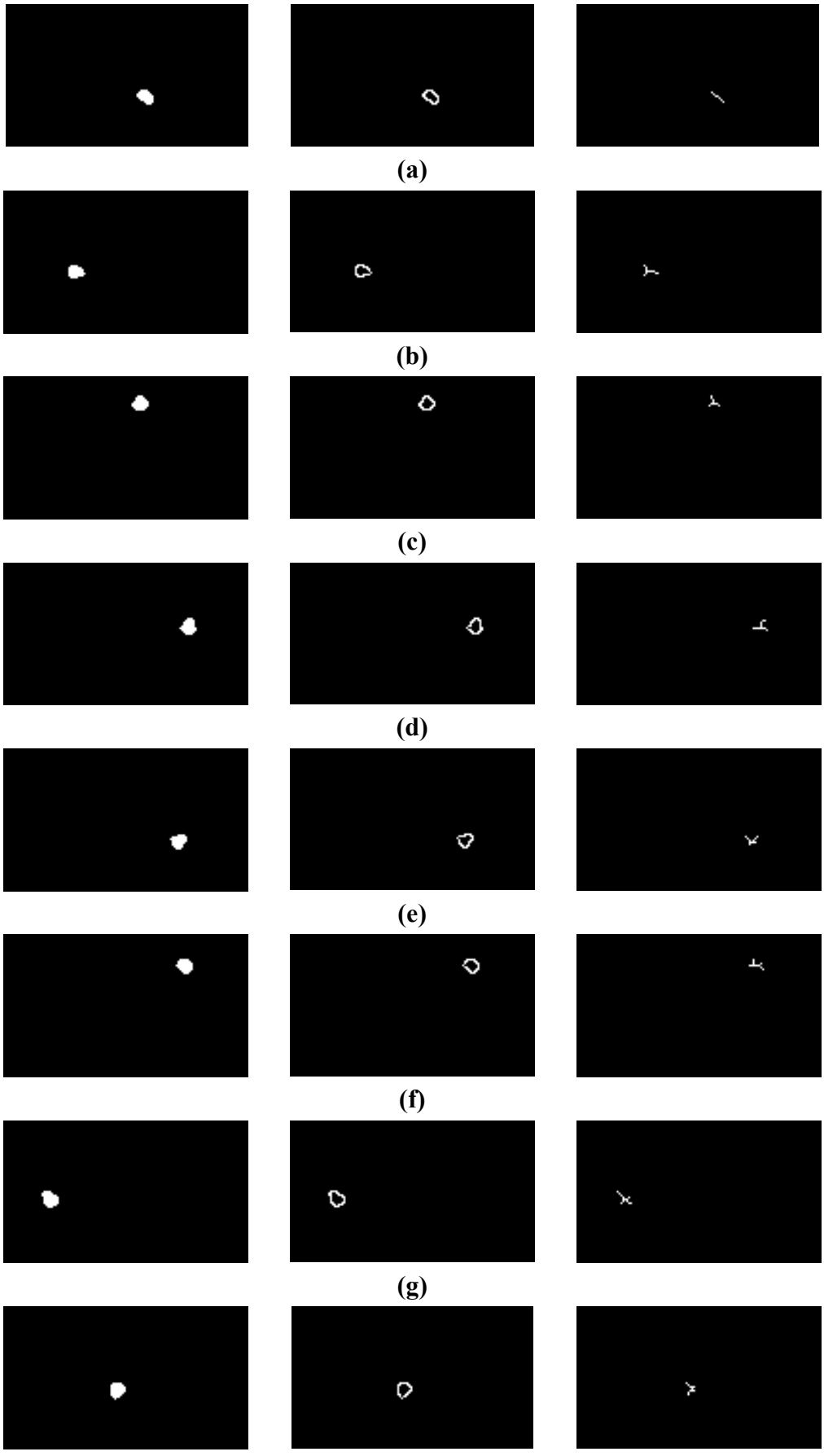
Önceki bölümlerde yürüyen dalganın yayılımı ve görüntü işlemeyle robot ve engel görüntüsünün çıkarımı anlatılmıştır. Bu bölümde anlatılan aşamalardan sonra robotun nasıl hareket edeceğinin kararlaştırılması anlatılacaktır. Robot hareketinin belirlenmesi de çalışma dönemi boyunca çeşitli değişikliklere uğramıştır. Yapılan ilk çalışmadan şu an uygulanan yönteme kadar tüm aşamalar verilecektir.

Robot şeklinin asıl resimden çıkarıldıktan sonra merkez noktası hesaplanır ve robotun platform üzerindeki yeri bulunur. Robot ilk hareketine başlamadan önce yön açısı bilinmelidir. Bir önceki bölümde robotun yön açısının hesaplanması görüntü işlemeyle belirlendiği söylenmiştir. Görüntü işlemeyle robotun yön açısının tam olarak hesaplanması ciddi bir sorundur. Doğru yön açısını bulmak için çeşitli yöntemler geliştirilmiştir. İlk olarak yön açısını robotun hareketini takip eden iki görüntüsündeki merkez noktalarından yola çıkılarak bulmak istenilmiştir. Başlangıçta robotun yeri tespit edildikten sonra yaklaşık 8 santimetrelik ilerlemesi sağlanmış ve yön açısı şimdiki ve başlangıçtaki merkez noktaları kullanarak hesaplamıştır. Böylelikle robotun gezinime başlamadan önce platformun üzerine nasıl konulduğu öğrenilmektedir. Bu ilk hareketin hangi yöne ve hangi açıyla yapacağının belirlenmesi için önemlidir. İlk hareketini takip eden hareketler için robotun anlık konum değeri ile önceki konumu arasındaki durumdan mevcut yönü bulunmaktadır. Alınan görüntü her zaman istenilen kalitede olmamaktadır. Bu, robot resminin tam olarak çıkarılamamasına dolayısıyla da merkez noktasının yanlış hesaplanmasına neden olmaktadır. Bu hataların bir araya gelmesiyle robotun anlık yön açısının yanlış hesaplanmasına neden olmaktadır. Robotun olduğundan farklı bir yönde olduğunun sisteme aktarılması, robotun yanlış yönlendirilmesine neden olmaktadır. Robot ileri yönde düz hareket ettiği durumlarda bile merkez noktası bir öncekinden geride hesaplanabilmektedir. Bu sistemin robotun ileri değil de kendi ekseni etrafında 180 derece döndüğünü zannetmesini neden olmaktadır. Bu durum robotun bir sonraki hareketin etkilemekte ve robotun beklenmeyen tepkiler vermesine neden olmaktadır. Bunun önüne geçebilmek sistemin alınan merkez koordinatlarının doğruluğunu sorgulayacak şekilde değiştirilmiştir.

Robotun bir önceki konumu ve şimdiki konumuna gelmek için yaptığı hareketten yola çıkılarak, hesaplanan robot merkezinin koordinatların doğruluğu

sorgulanmaktadır. Örneğin, robot 0 derece açıyla ileri gittiyse merkez koordinatları da önceki koordinatlarına göre kıyaslandığında x eksenin artış olması, y ekseninde ise değişiklik olmaması gerekmektedir. Hata tespit edildiği durumda merkez koordinatları olması gerektiği şekilde sistem tarafından düzeltilir ve robotun hareketi belirlenir. Alınan önleme rağmen çıkan sonuçlar her zaman istenilen düzeyde olmamıştır. Çözüm için sorunların temeli olan robot görüntüsünün asıl resimden çıkarılması üzerinde durulmuştur. Görüntü işleme algoritması tamamen değiştirilerek eskisine göre biraz daha uzun süren fakat daha kesin sonuç alabilen bir algoritma kullanılmıştır. Bu Bölüm 4.2. de anlatılan görüntü işleme metotlarından görüntü bölütleme yöntemidir. Böylelikle robot ve engel resimleri daha net alınmaktadır. Bu, robot merkez koordinatlarının hesaplanmasındaki hataları azaltmış fakat istenilen düzeye getirememiştir.

Robotun yönü merkez koordinatlarının dışında başka bir alternatif yoldan üretilmeye çalışılmış ve robotun görüntüsünün Matlabde iskelet yapısı çıkartılarak robotun yönü bulunmaya çalışılmıştır. Şekil 4.19 da robotun görüntüsü, alınan görüntünün sınırındaki sonlanma noktaları ve iskelet yapısı gösterilmiştir. Beklenen iskelet yapısı robotun her görüntüsü için Şekil 4.19-a daki gibi olmasıdır. Tek bir çizgiyle elde edilen iskelet yapısından yön rahatlıkla çıkartılabilir. Çıkan iskelet yapıları Şekil 4.19-b, Şekil 4.19-c ve Şekil 4.19-d deki gibi üç başlı olsa dahi robot yönü bulunabilirdi, fakat Şekil 4.19-e den Şekil 4.19-h e kadar olan görüntülerde dört başlı iskelet yapılarının gözlenmesi bu yöntemden vazgeçilmesine neden olmuştur. Bir sonraki yaklaşım robotun sahip olduğu geometrik yapısını kullanarak yönün bulunmasıdır. Görüntü işlemeden elde edilen robot görüntüsü dikdörtgensi bir görünüme sahiptir. Alınan görüntünün köşe noktaları belirlenip uzun kenarlarından geçen her bir doğrunun eğim ortalamasını alarak robotun yön açısı çıkartılabilir. Tek sorun robotun ön yüzünün hangi tarafa baktığının bu işlemle çıkartılamaması. Robotun x eksenine paralel konduğunu düşünüldüğünde görüntü işlemeden gelen robot yön açı değeri 0 derece olacaktır fakat robotun yüzü –x yönüne bakacak şekilde de x eksenine paralel konmuş olabilir. Bu durumda gelen açı değeri 0 derece değil 180 derece olmalıdır. Bu sorun da robot bir önceki hareketi ile şimdiki hareketi arasındaki ilişkiye bakılarak çözülmüştür. Eğer robotun merkez koordinatları bir öncekine göre x ekseninde azalma olmuşsa –x yönüne bakıyor demektir ve görüntü işlemeden gelen açı değeri bu bilgiye göre tekrar değerlendirilir.



Şekil 4.19: Robot resmi, sonlanma noktaları ve iskelet yapısı

Şekil 4.20 de yön açısının robotun geometrik şeklinden bulunma aşamaları gösterilmektedir. Şekil 4.20-a da platform üzerinde bulunan iki robotun resmi görülmektedir. Bunlardan biri takipçi robot diğeri kaçan robottur.

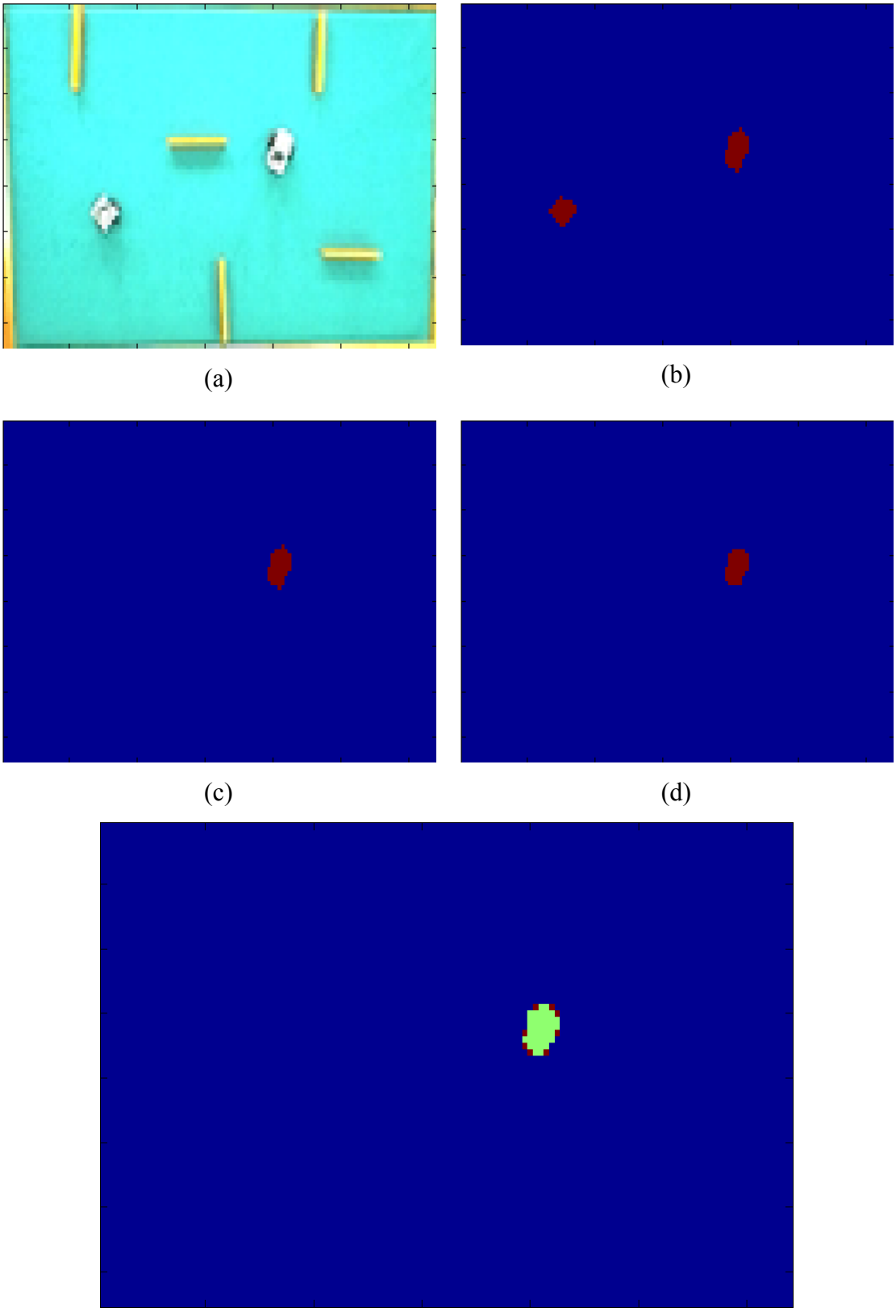
Görüntü bölütleme yöntemiyle elde edilen robot resimleri Şekil 4.20-b de verilmiştir. Robotları görüntü işlemede birbirinden kolay bir şekilde ayırt etmek için takipçi robotun daha bir yatay, kaçan robotun daha bir dik tasarımı sahip olduğu bir önceki bölümde anlatılmıştı. Bu özellikten faydalanarak takipçi robot kaçan robottan ayrılarak Şekil 4.20-c de gösterilmiştir. Dikdörtgensi bir yapıya sahip olduğundan birbirine paralel iki çizgiye bulabilmek için robotun köşelerinin belirlenmesi gerekmektedir. Bunun için robot görüntüsü 8 farklı şekilde taranır. Robotun en üst satırında ilk ve son sütunlarındaki pikseller, en alt satırında ilk ve son sütunlarındaki pikseller, en sol sütununda ilk ve son satırlarındaki pikseller ve en sağ sütunundaki ilk ve son satırlarındaki pikseller bulunur. Bu işlemin gerçekleştirilmesinden önce çıkıntı durumundaki sadece tek komşusu olan piksellerin elenmesi gerekmektedir. Şekil 4.20 de robot görüntüsünde ilk ve son satırdaki piksel görüldüğü üzere tek olarak çıkıntı durumundadır. Bu pikseller elenmezse, köşe bulma için 8 farklı taramada bulunan köşe sayısı 8 değil 6 olacaktır. Çünkü en üstteki piksel robotun en üst satırındaki hem ilk sütundaki piksel oluyor hem de en son satırdaki piksel oluyor. Yani iki köşe tek bir pikselde bulunuyor. Aynı durum en alttaki piksel için de geçerli en alt satırdaki ilk ve son sütündeki piksele bakıldığında tek bir piksel bulunacaktır. Bunu engellemek için Şekil 4.20-d görüldüğü üzere çıkıntılar yok edilir. Tarama yapıldıktan sonra bulunan 8 köşe Şekil 4.20-e de gösterilmiştir. Bundan sonraki aşamada birebirine yakın olan noktalar çiftler halinde bir araya getirilir ve ortalaması alınarak sekiz nokta değeri dörde düşürülür. Bu dört noktadan iki çizginin çizilmesi için herhangi bir nokta referans seçilir ve bu noktadan diğer üç noktaya olan uzaklıklara bakılır. En kısa ve en uzak mesafenin dışındaki ortanca nokta ile bu referans noktası çift seçilir. En kısa noktanın seçilmemesinin nedeni dikdörtgensi yapının kısa kenarını oluşturmamak, en uzun noktanın alınmamasının nedeni ise köşegen çizgisini seçmemektir. Referans noktayla eşleşme yapıldıktan sonra geriye kalan iki nokta birbirinin çifti seçilir. Bu iki çift noktadan çizilen iki çizginin eğimlerinin ortalaması alınarak robotun yön açısı hesaplanmış olur. Bu yöntem görüntü işlemeyle robot yönü bulmada gelinen en son noktadır. Bununla birlikte sistem istenilen performansta çalışmaktadır. Daha sonra Lego Mindstorm şirketinin

kullandığımız robotla uyumlu magnetik pusulayı çıkarması, bizi robotun yönünün görüntü işlemeyle bulmak yerine bu ürünü kullanmaya sevk etmiştir. Robotun yön bilgisini yeryüzündeki manyetik dalgalardan faydalanarak bulup sayısal olarak robotun giriş portlarına vermektedir. Buradan alınan sayısal yön bilgisi sisteme uyarlanarak robotun çalışması performansı arttırılmıştır.

Robotun yön bilgisi artık manyetik pusuladan gelen veriye tespit edilmektedir. Robotun nasıl ilerleyeceği ise dalga yayılımı sırasında, bütün hücrelerden ana hedefe giden en kısa yolu yönleriyle içeren vektör haritasına göre karar verilir. Şekil 4.21-a'da vektör haritasının oluşumu verilmiştir. Robotu hareket ettirmek için merkez noktası bulunur ve vektör haritasında merkezden 10 piksel uzaklıktaki yerel hedef tespit edilir. Varsayalım robotun merkez noktasının yeri Şekil 4.21-b'de yakınlaştırılmış vektör haritasında gösterildiği gibi olsun. Bulunduğu noktadan itibaren Şekil 4.21-c de gösterildiği gibi birer birer toplamda 10 adım gidilir ve Şekil 4.21-d'de gösterilen robotun hareket sonlanma noktası bulunur. Bu noktaya yerel hedef noktası da denir. Robotu her hareketinin amacı aslında yerel hedefe ulaşmaktır. Yerel hedefler üzerinden robotu ana hedefe taşınması sağlanmaktadır. Şekil 4.21-e'de robotun başlangıç noktasından yerel hedefe giden iki yol gösterilmiştir. Kesik çizgiyle gösterilen yol vektörsel yöndür.

Başlangıç noktasından yerel hedefe kuşbakışı gidiştir. Robotun tüm hareketi boyunca düzgün ve durmaksızın bir gezinim yapması için robotun bu vektörel yolu değil de Şekil 4.21-e'de gösterilen noktalı olarak gösterilen kavisli yolu izlemesi istenmiştir. Robotun gezinimi sırasında kontrol bilgisayarından gönderilen veriler tekerlekler için gerekli güç değerleridir.

Robotun bulunduğu noktadan yerel hedefe olan vektörsel yön ile robotun yön açısı arasındaki fark bize dönüş açısını vermektedir ki, yerel hedefe ulaşılmadan önce robot açısını dönüş açısı kadar değiştirmesi gerekir. Robot yön açısı ve yerel hedefe olan vektörsel açının değer aralığı [- 180°, 180°] arasında değişmektedir. Eğer dönüş açısı [-40°, 40°] aralığında ise, güç büyüklükleri robotun ileri yönde hareketi için yapılmış olan değer matrisinden alınmaktadır. Değer matrisi robotun farklı güç değerlerindeki hareketi gözlenerek oluşturulmuştur. Değer matrisi sadece dönüş açısının [-40°, 40°] aralığında olduğu zamanda kullanılmaktadır. Bu aralık elbette arttırılabilir, fakat bu robotun yerel hedefe vardığındaki son açıdaki sapmaya artıracaktır.



(a) (b)

(c) (d)

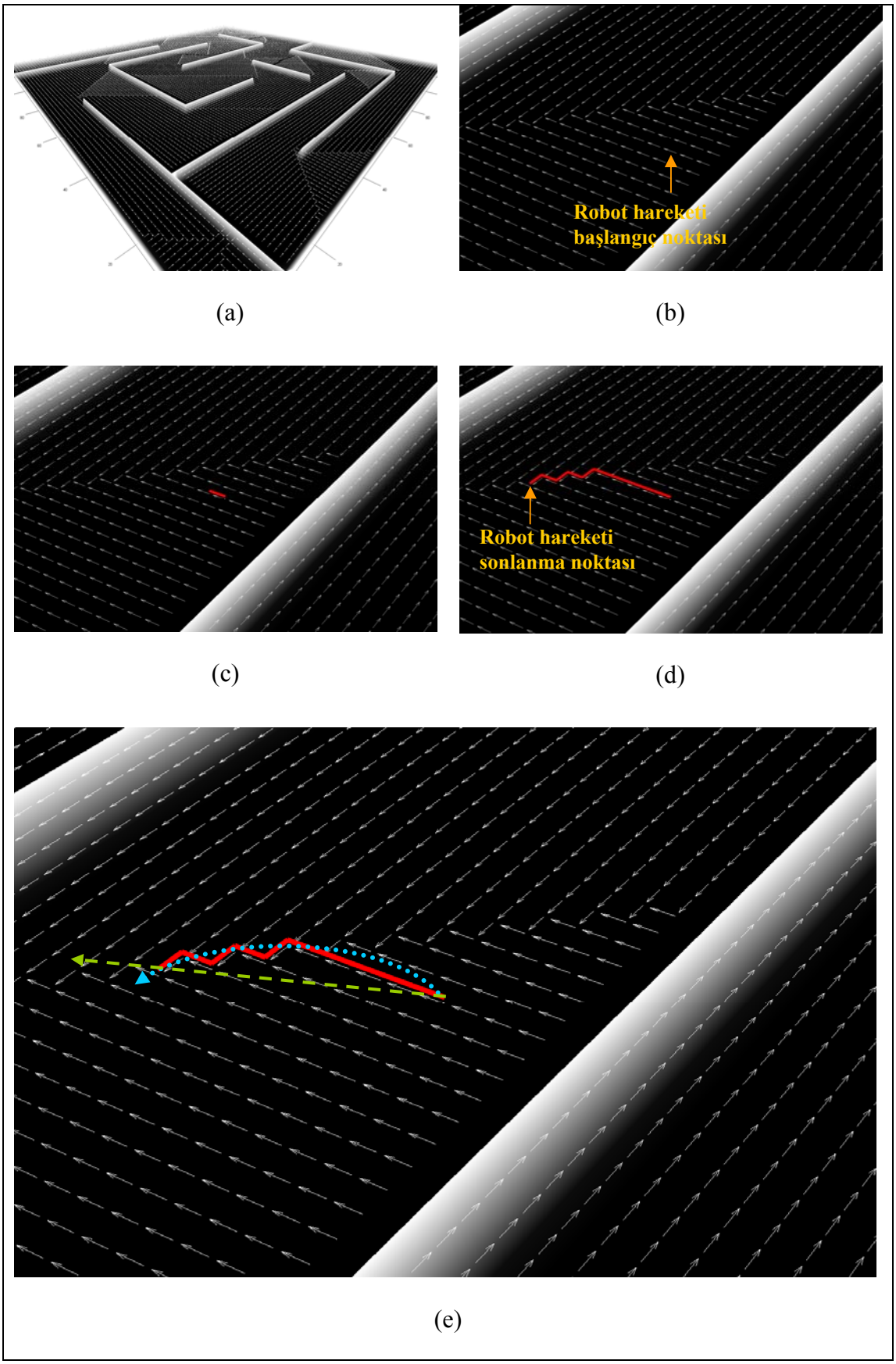
(e)

Şekil 4.20: Yön açısının robotun geometrik şeklinden bulunması

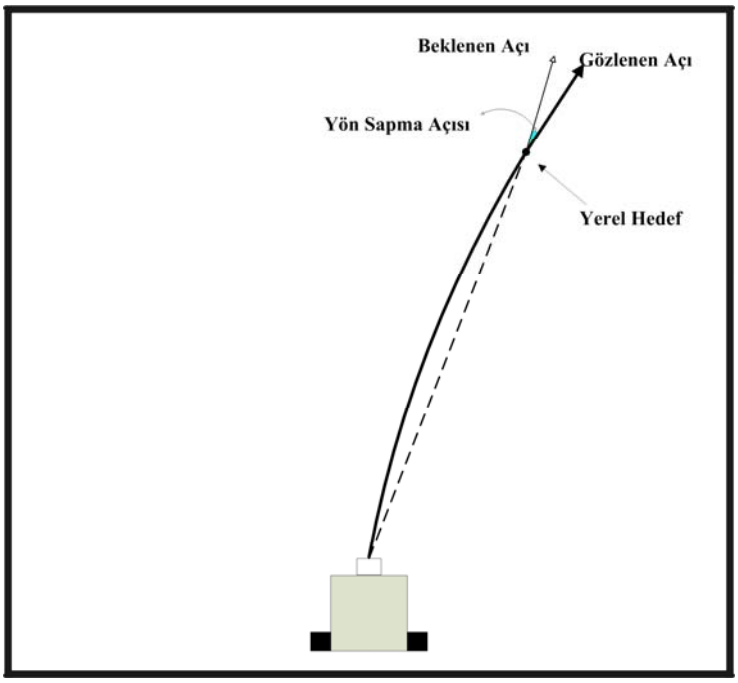
Şekil 4.21-e'de vektörel yön ile kavisli yönün sonlanma şekli gözükmektedir. Şekil 4.22'da değer matrisinin aralığının neden kısa tutulması gerektiği daha ayrıntılı

olarak gösterilmiştir. Şekil 4.22–(a)'da robot küçük dönüş açısıyla ve Şekil 4.22 – (b)'de büyük dönüş açısıyla ilerlemektedir. İki şekil incelendiğinde görülmüştür ki son açıdaki sapma dönüş açısına paralel olarak artmaktadır.

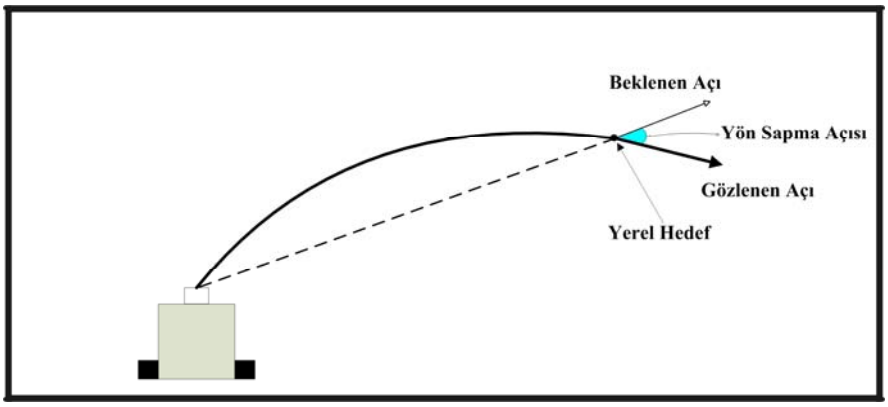
Dolayısıyla, değer matrisi aralığının küçük tutulması son açıdaki sapmayı azaltması ve robotu az hatayla ilerlemesi açısından fayda sağlayacaktır. Dönüş açısının mutlak değeri 40^0 büyük olduğu durumlarda robot dönüşümünü kendi ekseni etrafında tamamlayıp sonra ileri yöndeki hareketine devam etmektedir. Şekil 4.23'de dönüş bölgeleri gösterilmektedir. [-40°, 40°] aralığındaki dönüş açılarında robot ileri gidiş bölgesine denk düşmektedir ve güç büyüklüklerini değer matrisinden almaktadır. Matristen gücün nasıl alınacağı robot merkezinde yerel hedefe olan Öklid uzaklığı ve dönüş açısıyla belirlenmektedir. Mutlak dönüş açısının [41°, 130°] aralığında olduğu zaman, sistem tarafından tek motor için güç değeri hesaplanmaktadır. Bu durumda dönüş tek motorun hareketiyle sağlanmakta olup diğer motor durmaktadır. Mutlak dönüş açısının 130^0 büyük olduğu durumlarda iki motorda birbirinin ters doğrultuda sistemin hesapladığı süre boyunca dönmektedir.



Şekil 4.21: Robot yönünün tayini

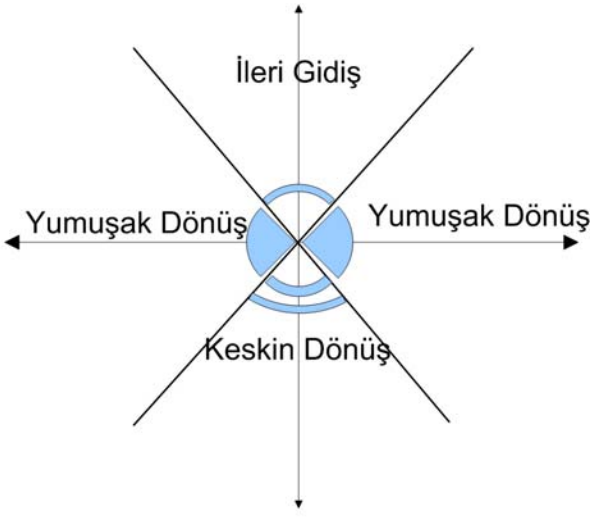


(a)



(b)

Şekil 4.22: (a) dar ve (b) geniş dönüş açıları için yön açısında sapma



Şekil 4.23 : Dönüş açısına bağlı olarak robot hareketinin bölgelere ayrılması

5. ROBOT GEZİNİM ALGORİTMASI

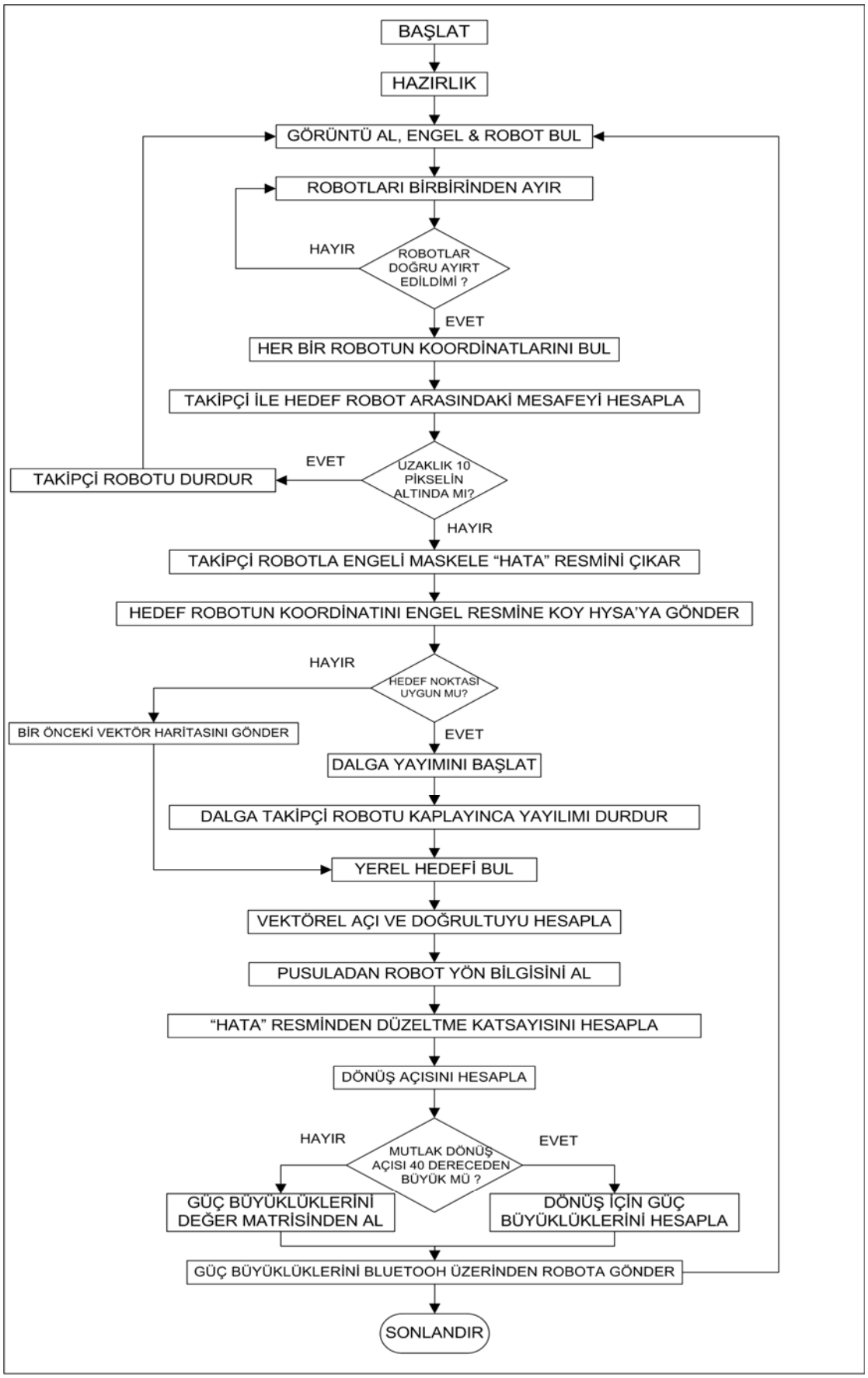
Önerilen takip algoritmasında hedef nesne ağ üzerinde bir hücre ile temsil edilmekte ve yürüyen dalganın başlangıç noktası bu hücre olmaktadır. Ortamdaki engeller de ağ üzerinde sabitlenmiş hücreler ile temsil edilmektedir. Bu yolla üretilen vektör haritasının takipçiyi her nerede olursa olsun hedefe götüreceği açıktır. Şekil 5.1'de gösterilen algoritma Matlab programının altında koşturulmaktadır. Sistemin çalıştırılmasıyla öncelikle genel (global) değişkenler tanımlanır ve sonra tavana yerleştirilmiş kameradan görüntü alabilmek için haberleşme kanalı açılır. Kameranın tanımlanmasından sonra robotun tanımlanmasına geçilir. Bluetooth haberleşmesi için gerekli com'lar açılır ve Matlab robota Bluetooth üzerinden bağlanılır(Ayrıntılı bilgi için alt Bölüm 4.1.1 e bakınız.). Görüntü işleme metodu olarak görüntü bölütleme yöntemi kullanıldığından örnek bir resim üzerinden robot, engel ve zemin ağırlıkları çıkartılır. Buraya kadar olan kısımlar hazırlık kısmıdır ve sadece programın ilk çalıştırılmasında yapılır. Sonsuz döngü içinde takip algoritması çalışırken bu işlemler bir daha yapılmayacak, böylelikle robot ve kamera bağlantılarını yapma, örnek resimden ağırlık çıkarma gibi uzun süren işlemlerle zaman kaybedilmeyecektir. Şekil 5.2'de gösterilen ara yüzle yapılan tüm adımlar takip edilebilmektedir. Bu aşamadan sonra program 'while' döngüsü içine girmekte ve ara yüze konulan "*sonlandır*" butonuna basılına kadar sistem bu döngünün içinde çalışmaktadır. Döngü içinde yapılan ilk işlem kameradan görüntü almak ve bu görüntüyü işlemektir. Kameradan YUV formatında ve 160x120 boyutunda alınan görüntü RGB formatında 128x96 boyutuna düşürülür. Şekil 4.12 de verilen resimde de gözüken alt taraftaki beyaz band kaldırılır ve görüntü 128x96 dan 128x75 e düşürülür. Görüntü işleme 128x75 lik görüntü üzerinden başlar. Görüntü bölütleme yöntemiyle en başta örnek resim üzerinden elde edilen robot, engel ve zemin ağırlıklarıyla anlık çekilen resimden robot, engel ve zemin resimleri çıkartılır. Engel ve robot resimleri birbirinden ayrılır ve ara yüzde gösterilir. Sistemde iki robot olduğundan elde edilen robot resmindeki iki robotun birbirinden ayrılması, hangisinin takipçi hangisinin hedef robot olduğunun belirlenmesi gerekir. Görüntü işleme bölümünde anlatıldığı gibi robotları ayırt etmek için hedef robot dik, takipçi robot yatay olarak tasarlanmıştır. Takipçi robot yatay tasarımından dolayı görüntü üzerindeki izdüşüm alanı hedef robota göre

fazladır. Buradan yola çıkarak robot resminden takipçi robot ve hedef robot birbirinden ayrılır ve merkez koordinatları bulunur. İzdüşüm alanına göre yapılan bu ayırmada hataya yer vermemek için çıkan sonuçlar kontrol edilir. Bunun için takipçi robotun güncel merkez koordinatları ile eski merkez koordinatları ve hedef robotun güncel koordinatları arasındaki uzaklığa bakılır. Robotlar doğru ayırt edilmişse takipçi robotun güncel ile eski koordinatları arasındaki uzaklık takipçi robot ile olan uzaklıktan daha kısa çıkar. Eğer sonuç tam tersi çıkarsa robotlar yanlış ayırt edilmiş demektir ve düzeltilir. Takipçi robot ile hedef robot arasındaki mesafeye bakılarak aralarındaki mesafenin 10 pikselin altında olup olmadığına bakılır. İki robot arasındaki mesafe 10 pikselin altına düştüğü zaman takipçi robotun hedef robotu yakaladığı kabul edilmektedir ve aradaki mesafe 10 pikselin üzerine çıkana kadar takipçi robot durmakta ve sürekli bir şekilde platformun görüntü işlenerek robotlar arasındaki mesafe kontrol edilmektedir. Aradaki mesafe 10 pikselin üzerindeyse hedef robot ile genişletilmiş engel resmi maskelenerek hedef robotun engelin içine girip girmediğine bakılır. Maskeleme sonucunda çıkan resim hata resmi olarak kaydedilir ve daha sonra robot döneceği açı hesaplanırken düzeltme katsayısı hesaplanmasında kullanılacaktır.

Görüntü işleme ile ilgili tüm işlemler tamamlandıktan sonra dalga yayma işlemine geçilir. Bunun için 128x75 lik genişletilmiş engel resmi 128x128 hale getirilir. Hedef robotun yeri bu resme işlenir ve tasarlanan hücresel yapay sinir ağına gönderilir. Burada hedef noktası kaynak nokta olarak kabul edilir ve bu noktadan başlayarak yürüyen dalgalar yayılır. Çıktı olarak tüm hücrelerden kaynak noktaya giden en kısa yolu veren vektör haritası alınır. Vektör haritasının güncellenme sıklığını arttırabilmek için yürüyen dalga, ağdaki en son hücreye ulaşana kadar değil, takipçiyi temsil eden hücreye kadar yayılmıştır. Bu sayede güncelleme periyodu, hedef ile takipçi arasındaki uzaklıkla doğru orantılanmış olur**.** Hedef robot kontrol bilgisayarından klavyenin ok tuşları ile Matlab üzerinden kontrol edilmektedir. Kullanıcı tarafından gezdirilen hedef robot bazen genişletilmiş engel sınırlarının içine girebilmektedir. Bu durumda kaynak nokta engelin içinde olduğundan dalganın yayılması ve dolayısıyla en kısa yolu veren vektör haritasını almak mümkün değildir. Bu tür durumlarda hedef robot bir önceki konumundan çok fazla uzaklaşmadığı kabul edilerek önceki vektör haritası sisteme çıktı olarak veriliyor. Böylelikle hedef robotun engele girdiği durumlarda sistemin hata vererek döngüden çıkması

engellenebilmektedir. En kısa yolu veren vektör haritası alındıktan sonra takipçi robotun koordinatları bu harita üzerinde bulunarak 10 piksel ötedeki yerel hedef bulunur. Bir önceki bölümde bu durum ayrıntılı açıklanmış ve Şekil 4.21'de gösterilmiştir. Amaç kaynak noktaya(hedef robota) yerel hedefler üzerinden gitmektir. Yerel hedef noktaları bulunduktan sonra takipçi robot ile arasındaki uzaklık ve vektörel açı hesaplanır. Manyetik pusuladan hedef robotun açı bilgisi alınır. Gelen veri 0 -360 derece arasındadır. Alınan bu veri [-180,180] arasına dönüştürülür. Aynı zamanda bölüm 4.3'de belirtilen robotun hangi yöne baktığı sorunu da giderilerek robotun gerçek açısı bulunur.

Bundaki sonraki aşama takipçi robotun yerel hedefe yönlendirilmesidir. Bunun için takipçi robotun yön açısından vektörel açı çıkartılarak robotun dönmesi gereken dönme açısı bulunur. Bu dönme açısına göre robotun yapacağı manevra belirlenecektir. Bunun nasıl belirlendiği ayrıntılı olarak bölüm 4.3'robot hareketinin belirlenmesi konusunda ayrıntılı olarak açıklanmıştır. Bu dönme açısı robotun yapacağı son hareketi belirlendiği için manevra belirlenmeden önce dönme açısına hata resmini kullanarak hesaplanmış düzeltme katsayısı eklenmektedir. Bu düzeltme katsayısı robotun engele daha fazla girmesini engellediği gibi takipçi robotun engelini içersinden çıkmasını da sağlamaktadır. Robota gidecek güç değerleri belirlendikten sonra robota Bluetooth üzerinden veriler gönderilir. Daha sonra tekrar kameradan görüntü alımına, robot ve engellerin bulunmasını geçilir. Robotlar arasındaki mesafeye bakılarak yakalamanın tamamlanıp tamamlanmadığına bakılır. Tüm bu işlemler ara yüzdeki sonlandır komutu gelene kadar devam eder.



Şekil 5.1: Robot gezinim algoritması

Şekil 5.2: Tasarlanan Arayüz

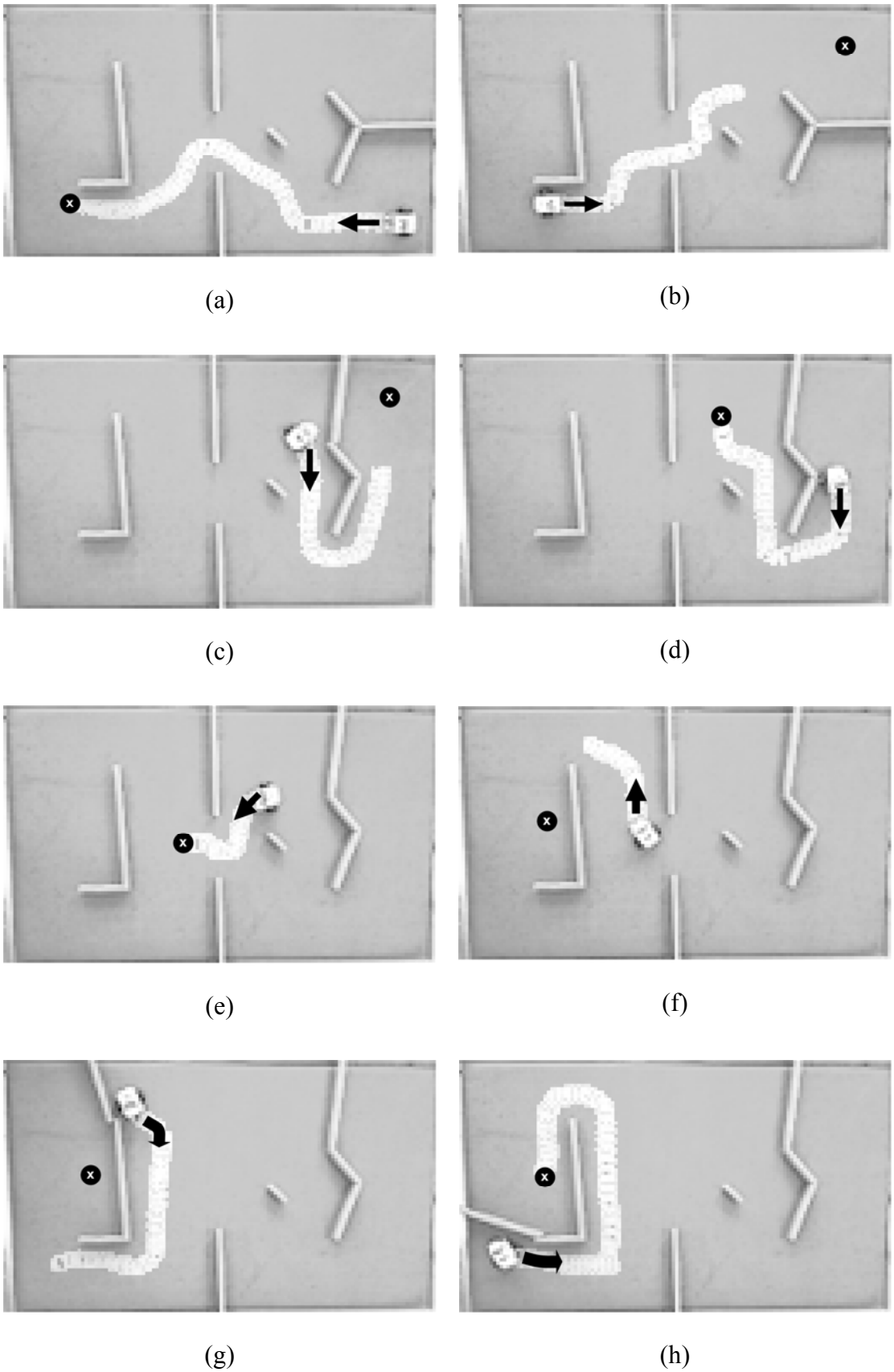
6. DÜZENEK ÜZERİNDE UYGULAMALAR

Bu çalışmayla, dinamik ortamdaki robotun yürüyen dalgalar kullanarak en kısa yoldan gezinimi başarıyla sağlanmıştır. Tüm sistemin tek bir yazılım tarafından kontrol edilmesine rağmen dalga yayılımı yeterince hızlı gerçekleşmektedir. Şekil 6.1'de tek robotun kullanıcı tarafından belirtile hedef noktalara gitmesini gösteren bir senaryo gösterilmektedir [20]. Şekil 6.2'de ise çalışmaların ilerlemesiyle geçilen kovalamaca senaryosu verilmiştir [21]. Burada farklı durumlar için takipçi robotun hedef robotu yakalaması gösterilmiştir. Şekil 6.1 ve Şekil 6.2'de beyazla gösterilen yollar robotların gezinim sırasında izlediği yolları göstermektedir. Şekillerden görüldüğü üzere takipçi robot, hedef nokta veya robotun anlık konumuna göre en kısa yolu izlemektedir. Takipçi robotun kovalamayı başarılı bir şekilde gerçekleştirmesi için hedef robotun veya noktanın yerinin belirlenmesi dalgaların yayılması ve uygun hız değerlerinin hesaplanıp robotlara gönderilmesi gibi işlemlerin tamamının 1 saniyenin altında gerçekleşmesi gerektiği tespit edilmiştir. Aksi durumda tepki süresi artmakta ve bu durum takipçi robotun engele çarpmasına veya hedef robota giden en kısa yolu takip edememesine neden olmaktadır. Genel olarak görüntünün islenmesi 0.25 saniye sürmektedir. Dalgaların yayılıp en kısa yolun bulunması 0.25 saniye almaktadır. Yayılma zamanı platform üzerindeki engellerin konumuna ve robot ile hedef noktasına ulan uzaklığa bağlı olarak değişmektedir. Takipçideki pusuladan yön bilgisinin okunması 0.10 saniye, takipçi robot için uygun güç değerlerinin hesaplanıp gönderilmesi 0.35 saniye sürmektedir. Bu değerler yaklaşık olup çalışma anında süreler değişebilmektedir. Ortalama olarak hedef takibi algoritmasının döngü periyodu 0.6 ile 1 saniye arasında değişmektedir.

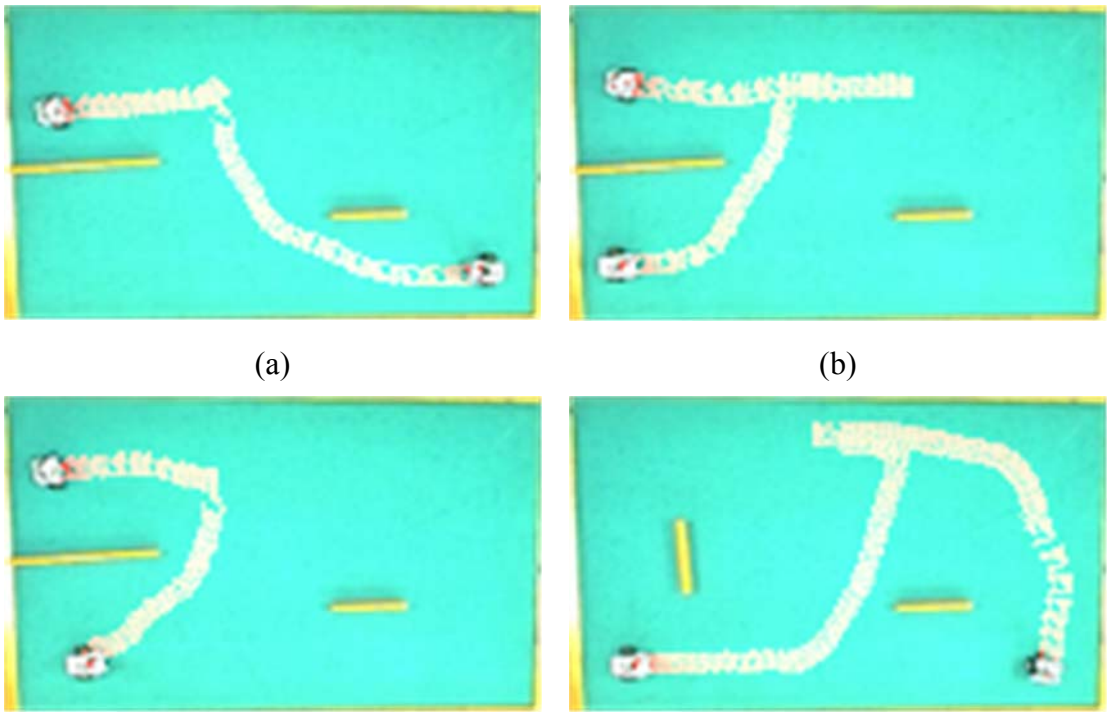
Şekil 6.1'de tek robotun kullanıcı tarafından bilgisayardaki ara yüzden girilen hedef noktayı takip etmesini göstermektedir. Şekil 6.1–(a)'da robot durmaksızın ileriye giderek ana hedefe ulaşmakta ve köşelerde yumuşak dönüşler yapmaktadır. Sadece büyük dönüş açılarında robot kendi ekseni etrafında dönmek için durmakta, dönüşünü tamamladıktan sonra ilerlemeye devam etmektedir. Robotun Şekil 6.1–(a)'daki hedefine ulaşmadan önce ana hedef değiştirildiğinde robot durur, geri döner ve Şekil 6.1–(b)'de bulunan yeni hedefine gider. Robotun dinamik ortamdaki

davranışı Şekil 6.1–(b)'den Şekil 6.1-(c)'ye geçişte gözükmektedir. Şekil 6.1-(c)'de robot aynı hedefe yeni kısa çıkartılmış kısa yoldan gitmektedir. Ana hedefin değiştirildiğinde robotun yolunu değiştirdiğini Şekil 6.1–(d) ve Şekil 6.1–(e)'de gözükmektedir. Robotun hedefe en kısa yoldan gitmeye çalıştığı Şekil 6.1 –(f)'de açıkça gözükmektedir. Robotun dinamik ortamdaki değişikliğe gösterdiği tepki Şekil 6.1–(g) ve Şekil 6.1–(h)'da gözükmektedir.

Robotun ana hedefe doğru olan hareketinde bazı durumlarda engele çarptığı gözlemlenmiştir. Bunun iki nedeni vardır. Birincisi robota gönderilen verinin yineleme sıklığının sınır değerin üzerine çıkmasıdır. Güncelleme sıklığının 0.7 saniye olduğu zaman robotun başarılı bir şekilde hareketine devam ettiği görülmüştür. Bu yüzden 0.7 saniye sınır değeri olarak kabul edilmiştir. Dalga yayımı hedef noktadan başlar ve robotun merkezine ulaştıktan yirmi iterasyon sonra biter. Bunun anlamı robot ile hedef nokta arasındaki uzaklık arttıkça dalga yayım süresinin uzamasıdır. Bununla birlikte platformda bulunan engellerin sayısı ve uzunluğu da dalga yayım süresini etkilemektedir. Basit bir ortamda en kısa yol haritasını çıkarmak az zaman alırken dolambaçlı bir ortamın en kısa yol haritasını çıkarmak daha fazla zaman almaktadır. İkinci sorun ise görüntü işleme ile robotun yönünün tam olarak çıkarılamamasıdır. Yönün yanlış hesaplanması robotun yanlış yönlendirilmesine ve robotun engele çarpmasına veya platformun dışına çıkmasına neden olmaktadır. Yön açısının doğru hesaplanması robot görüntüsünün doğru çıkarılmasına bağlıdır. Bu sorunun üstünden gelmek için görüntü işlemedeki algoritma değiştirildi. Manyetik pusulanın alınmasıyla bu sorun ortadan kalkmış oldu fakat manyetik pusulanın getirdiği handikap pusuladan alınan verinin hızlı olmaması ve çalışma anında sürekli değişim göstermesidir. 0.3 saniye de pusulanın yön bilgisi alındığı görülmüş 0.1 saniyenin altında da pusuladan veri girişinin olduğu gözlemlenmiştir. Yine de oluşabilecek çarpma olaylarının önüne geçmek için görüntü işleme sırasında robot resmi genişletilmiş engel resmiyle maskelenerek robotun engelin içine girdiği kısım olup olmadığına bakılıyor. Daha sonra bu kısmın robotun hangi yönüne denk geldiği ve kaç pikselden oluştuğuna bakılarak bir sonraki adımda robotun yapacağı dönüş açısını değiştirecek katsayılar hesaplanır. Bu katsayılarla robotun engele çarpma durumunu en aza indirilmiştir.



Şekil 6.1: Tek robot için basit bir gezinim senaryosu



(a) (b)

(c) (d)

Şekil 6.2: İki robotlu takip senaryosu

7. SONUÇLAR

Bu çalışmayla tasarlanan GO-HYSA üzerinde yürüyen dalgalar yayılarak en kısa yol bulma probleminin çözümlenmesi başarıyla gerçeklenmiştir.

Benzetimi yapılan GO-HYSA üzerinde sağlıklar sonuçlar alınmasının ardından ağın gerçeklenmesine gidilmiş ve gerçek ortamda kurulan labirent benzeri ortamda robotların hareketi sağlanmıştır. Sistem tamamen dinamikleştirilmiş böylelikle değişen ortama robotların hızlı tepki vermesiyle platform üzerinde düzgün ve akıcı dolanım sağlanmıştır. Takip algoritmasının sürekli olarak geliştirilmesiyle tek robotlu gezinimden iki robotlu kaçma kovalamaya geçilmiştir. Kaçan robot kullanıcı tarafından klavye ile kontrol ettirilirken takipçi robot sistemden gelen en kısa yol bilgisine göre hareket etmektedir. Tüm sistem bilgisayar ortamında çalıştırılmasına rağmen yaklaşık 0.7 saniyede takipçi robotun yön bilgisini göndererek platform üzerinde düzgün, duraksamadan hareket etmektedir. Matlab altında yapılan kodlamaların sayısal devre tasarımı yapılarak FPGA içine alındığında sistem daha yüksek hızda çalışarak robotun gezinimi sırasında dinamik ortamdaki değişikliklere daha hızlı cevap verecektir.

Önerilen yöntem, kuşbakışı ortam bilgisi ve hedef koordinatları bilinen tüm takip senaryolarında örneğin askeri uygulamalarda, trafik çözümlemelerinde ve her türlü yol saptama (routing) problemi çözümlerinde kullanılabilir.

KAYNAKLAR

[1] **Chua L. and Yang L.,** 1988. "Cellular neural networks: theory," *Circuits and Systems, IEEE Transactions on,* vol. 35, no. 10, pp. 1257–1272.

[2] **Chua L. and Yang L.,** 1988. "Cellular neural networks: applications,"*Circuits and Systems, IEEE Transactions on,* vol. 35, no. 10, pp. 1273–1290.

[3] **Chua L. and Roska T.**, "The CNN paradigm," *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, vol. 40,no. 3, pp. 147–156, Mar 1993.

[4] **Roska B. and Werblin F. S.**, "Vertical interactions across ten parallel,stacked representations in the mammalian retina," Nature, vol. 410, p.583, 2001.

[5] **Yalcin M.,** 2008. "A simple programmable autowave generator network for wave computing applications," *IEEE Transactions on Circuits and Systems II-Express Briefs,* vol. 55, no. 11, pp. 1173–1177.

[6] **Yeniceri R. and Yalcin M.,** 2008. "An implementation of 2D locally coupled relaxation oscillators on an FPGA for real-time autowave generation," *11th International Workshop on Cellular Neural Networks and their Applications, CNNA 2008,* pp. 29–33.

[7] **Yeniçeri R.,** 2009. "Yol Bulma Uygulamaları İçin Bir Hücresel Yapay Sinir Ağının Sayısal Tasarımı Ve Gerçeklenmesi" İ*stanbul Teknik Üniversitesi Fen Bilimleri Enstitüsü Yüksek LisansTezi*, İstanbul, Türkiye.

[8] **Yeniceri R. and Yalcin M.,** 2009. "Path planning on cellular nonlinear network using active wave computing technique*," Proceedings of SPIE Europe Microtechnologies for the New Millenium Symposium,*vol. 7365.

[9] **Adamatzky A., Arena P., Basile A., Carmona-Galan R., Costello B., Fortuna L., Frasca M. and Rodriguez-Vazquez A.,** 2004. "Reactiondiffusion navigation robot control: from chemical to VLSI analogic processors," *Circuits and Systems I: Regular Papers, IEEE Transactions on,* vol. 51, pp. 926–938.

[10] **Arena P., Basile A., Fortuna L. and Frasca M.,** 2004. "CNN wave based computation for robot navigation planning," *in Proc. Of the 2004 IEEE ISCAS,* Toronto, Canada.

[11] **Arena P., Fortuna L., Frasca M., Vagliasindi G. and Basile A.,** 2005. "CNN wave based computation for robot navigation on ACE16K," *Circuits and Systems, 2005. ISCAS 2005. IEEE Internation l Symposium on,* vol. 6, pp. 5818–5821.

[12] **Arena P., De Fiore S., Fortuna L. and Patane L.,** 2008. "Perception-action map learning in controlled multiscroll systems applied to robot navigation," *CHAOS,* vol. 18, p. 043119.

[13] **Ito K., Hiratsuka M., Aoki T. and Higuchi T.,** 2006. "A shortest path search algorithm using an excitable digital reaction-diffusion system," *IEICE Transactions on Fundamentals of Electronics Communications and Computer Sciences,* vol. E89A, pp. 735–743.

[14] **Ito K., Hiratsuka M., Aoki T. and Higuchi T.,** 2006. "A Shortest Path Search Algorithm Using an Exicitable Digital Reaction-Diffusion System," *IEICE Trans. Fundamentals,* vol. E89-A, no. 3.

[15] **Gavrilut I., Tiponut V. and Gacsadi A.,** 2006. "Path Planning of Mobile Robots by Using Cellular Neural Networks", *10th International Workshop on Cellular Neural Networks and their Applications, CNNA 2008,* Istanbul, Turkey.

[16] **Hiratsuka M., Ito K., Aoki T. and Higuchi T.,** 2008. "Shorthest Path Search Using a Reaction-Diffusion Processor," *Int. Journal of Unconventional Computing,* vol. 4, pp. 113-123.

[17] **Roska T., Chua L.**, 1993. "The CNN Universal Machine: An Analogic Array Computer," *IEEE Trans. Circuits Syst. II,* vol. 40, pp. 163-173.

[18] **Nagy Z. and Szolgay P.,** 2003. "Configurable multilayer CNN-UM emulator on FPGA," *IEEE Transactions on Circuits and Systems I:Fundamental Theory and Applications,* vol. 50, no. 6, pp. 774–778.

[19] **Kayaer K., Tavsanoglu V.,** 2008. "A new approach to emulate CNN on FPGAs for real time video processing," *11th Int.l Workshop on Cellular Neural Networks and their Applications, CNNA 2008,* pp. 23–28.

[20] **Kilic V., Yeniceri R. and Yalcin M.E**., " A New Active Wave Computing Based Real Time Mobile Robot Navigation Algorithm for Dynamic Environment*," in Proc.of the 12th IEEE International Workshop on Cellular Neural Networks and their Applications* (CNNA 2010), Berkeley, California, USA, Feb. 3-5, 2010, pp 1-6,

[21] **Yeniceri R., Kilic V. and Yalcin M.E**., "Uzay Zaman Dalgalarina Dayali Yeni Bir Hedef Takip Algoritmasi (A New Spatio-temporal Wave Based Target Tracking Algorithm)," *18. Sinyal Isleme ve Iletisim Uygulamalari Kurultayi* (SIU 2010), Diyarbakir, Turkey, April 22-24, 2010, pp. 562-565 (In Turkish).

[22] **Günay E.**,2005, "Voltaj Ve Akım-Modlu Olarak Tasarlanan Hücresel Sinir Ağları Ve Uygulamaları" *Erciyes Üniversitesi Fen Bilimleri Enstitüsü Doktora Tezi*, Kayseri Türkiye.

[23] **Yalcin M., Suykens J. and Vanderwalle J.,** 2004. "Experimental Observations of Autowaves on the ACE16k CNN Chip", *Proc. Of the 8th IEEE Int.Workshop on CNNA,* Budapest, Hungary.

EKLER

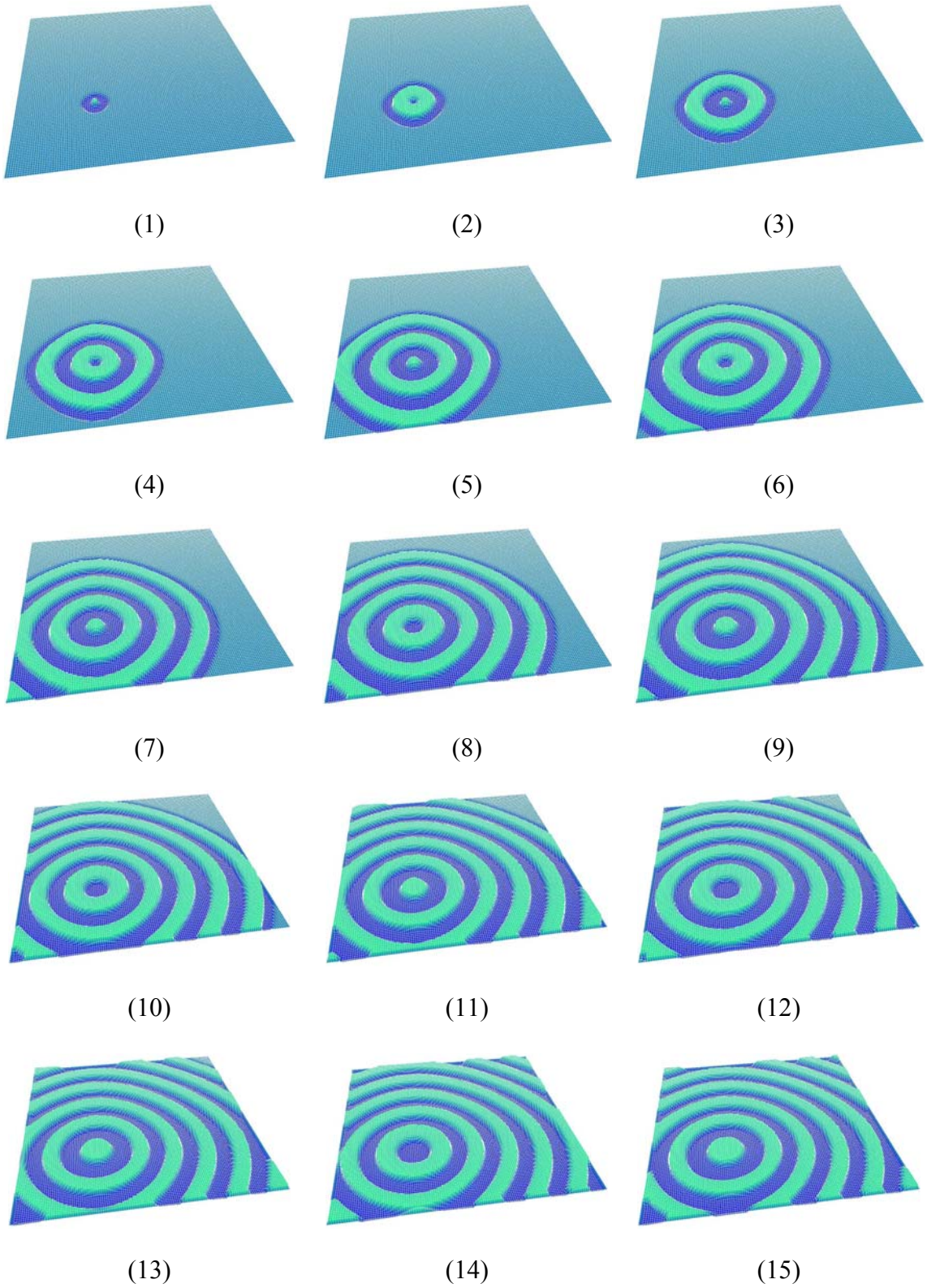
EK A.1 : Oto dalga yayılımı

EK A.2 : Spiral dalga yayılımı

EK A.3 : Yürüyen dalgaların yayılımı

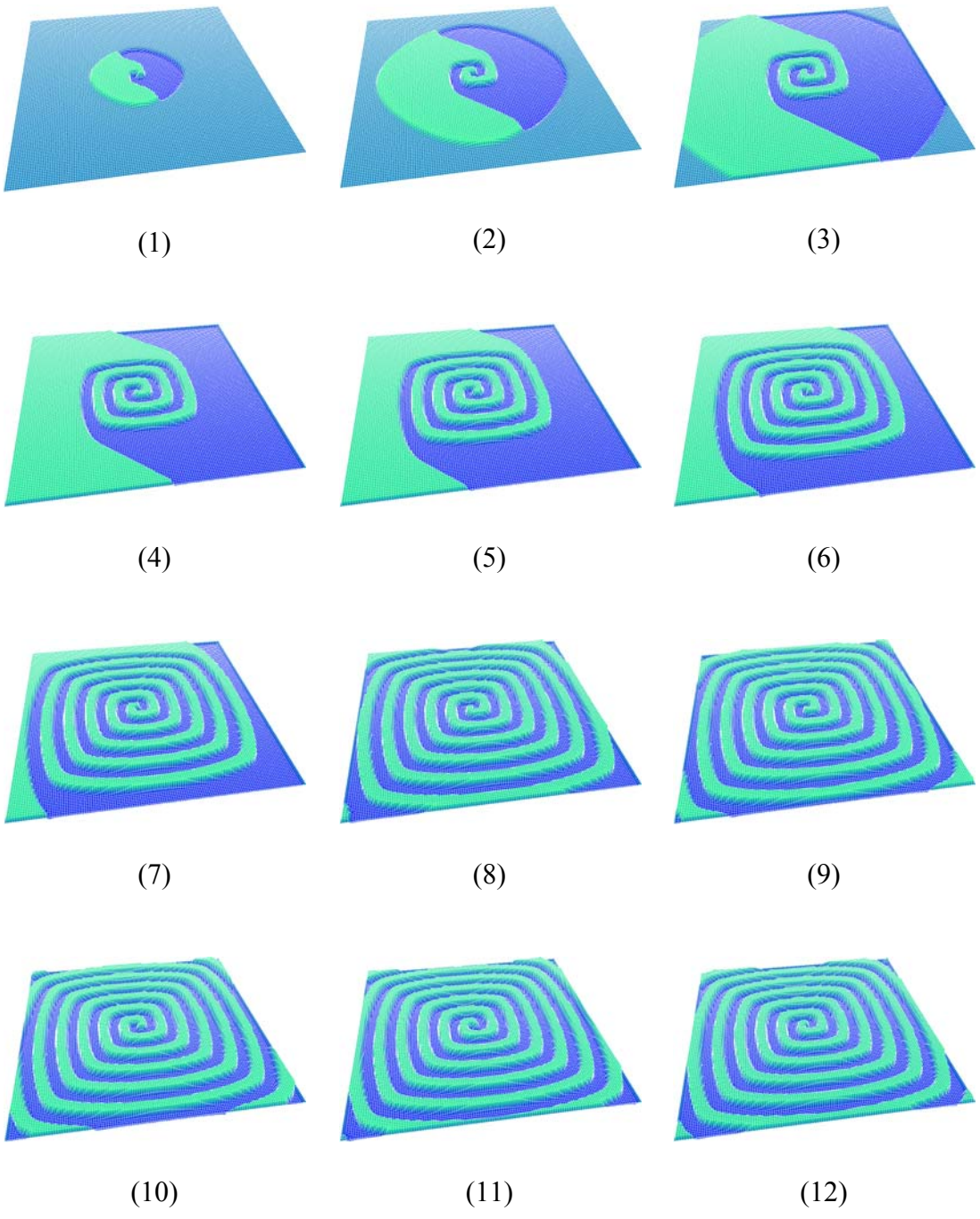
EK A.4 : Sabit hücreli ortamda yürüyen dalganın yayılımı

EK A.1



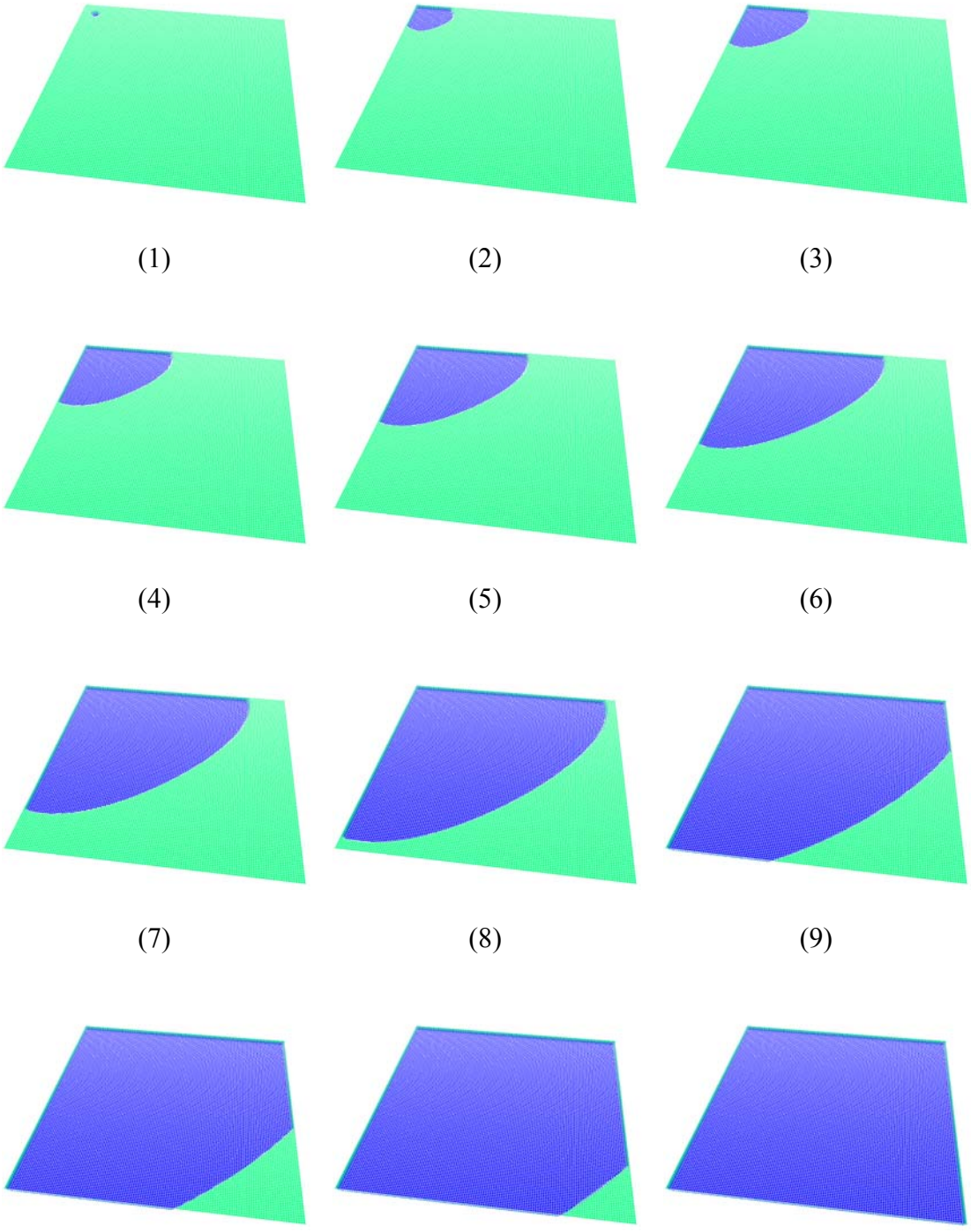
Şekil A.1: Oto dalga yayılımı

EK A.2



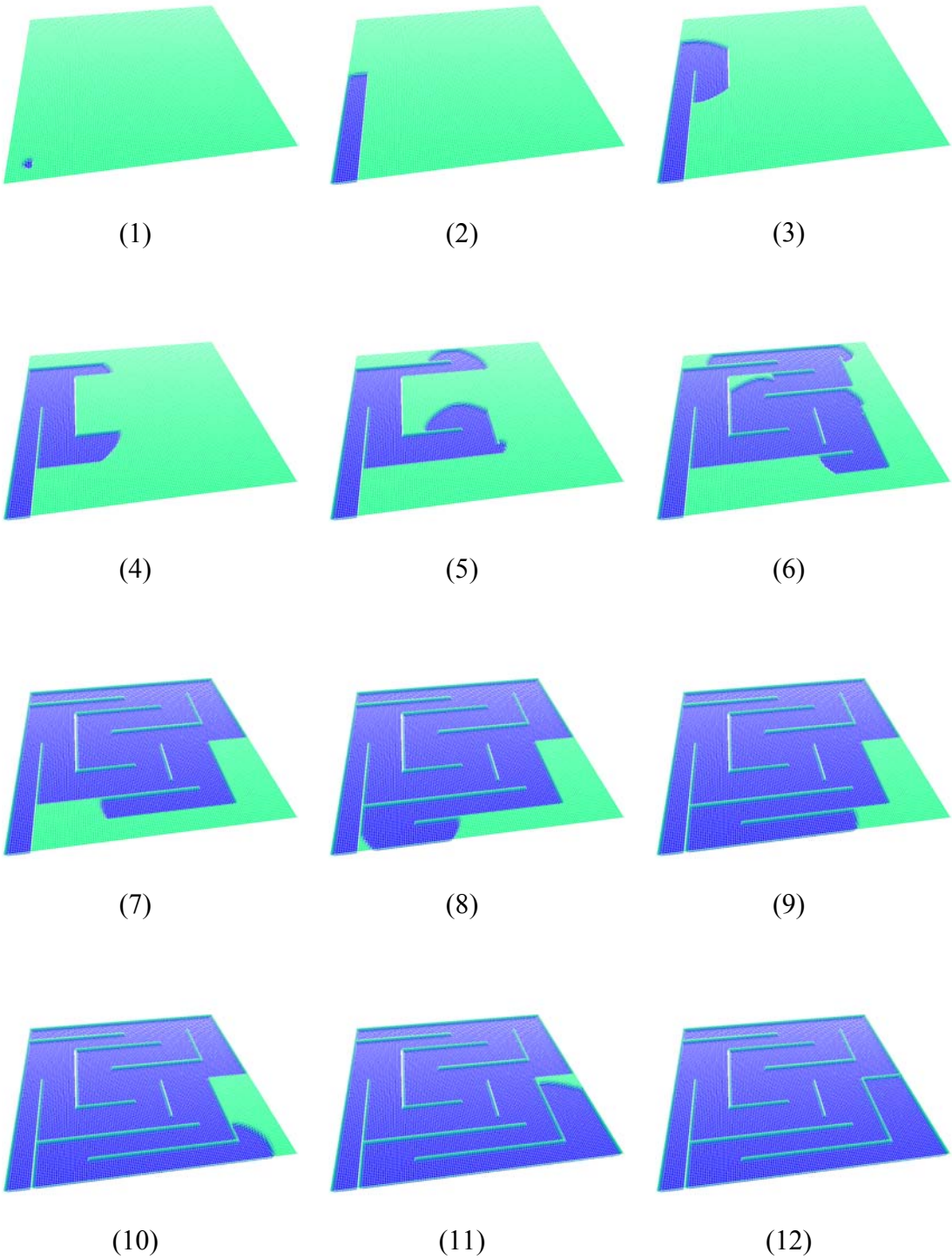
Şekil A.2: Spiral dalga yayılımı

EK A.3



Şekil A.3: Yürüyen dalgaların yayılımı

EK A.4



Şekil A.4: Sabit hücreli ortamda yürüyen dalganın yayılımı

ÖZGEÇMİŞ

Ad Soyad: Volkan KILÇ

Doğum Yeri ve Tarihi: İzmir, 1985

Adres: Bahçeli evler mah. Bahadır konakları sitesi Billur sok B Blok No: 16 daire: 39 Tepebaşı/ Eskişehir, Türkiye

Lisans Üniversite: Anadolu Üniversitesi, Elektrik-Elekktronik Müh.

Yayın Listesi:

- **Kilic V**., Yeniceri R. and Yalcin M.E., " A New Active Wave Computing Based Real Time Mobile Robot Navigation Algorithm for Dynamic Environment," in Proc.of the 12th IEEE International Workshop on Cellular Neural Networks and their Applications (CNNA 2010), Berkeley, California, USA, Feb. 3-5, 2010, pp 1-6, DOI: 10.1109/CNNA.2010.5430279.

- Yeniçeri R., **Kılıç V**., Yalçın M. E., "An On-line Test Setup of CNN Based Real-time Mobile Robot Navigation Application", Proc. of 12th International Wokshop on Cellular Nanoscale Networks and Applications (CNNA 2010), p. 1-1, Berkeley, USA, February 3-5, 2010.

- Yeniceri R., **Kilic V.** and Yalcin M.E., "Uzay Zaman Dalgalarina Dayali Yeni Bir Hedef Takip Algoritmasi (A New Spatio-temporal Wave Based Target Tracking Algorithm)," 18. Sinyal Isleme ve Iletisim Uygulamalari Kurultayi (SIU 2010), Diyarbakir, Turkey, April 22-24, 2010, pp. 562-565 (In Turkish).