T.C.
KONYA TEKNİK ÜNİVERSİTESİ
LİSANSÜSTÜ EĞİTİM ENSTİTÜSÜ

MOBİL ROBOTLAR İÇİN OPTİMUM YOL BULMA

Muhammed Esat DERE

YÜKSEK LİSANS

Elektrik – Elektronik Mühendisliği Anabilim Dalı

Temmuz-2019
KONYA

TEZ KABUL VE ONAYI

Muhammed Esat DERE tarafından hazırlanan "Mobil Robotlar için Optimum Yol Bulma" adlı tez çalışması 01/07/2019 tarihinde aşağıdaki jüri tarafından oy birliği ile Konya Teknik Üniversitesi Lisansüstü Eğitim Enstitüsü Elektrik-Elektronik Mühendisliği Anabilim Dalı'nda YÜKSEK LİSANS TEZİ olarak kabul edilmiştir.

Jüri Üyeleri	**İmza**
Başkan Prof. Dr. Cemil SUNGUR	.....
Danışman Dr. Öğr. Üyesi Akif DURDU	.....
Üye Dr. Öğr. Üyesi Burak YILMAZ	.....

Yukarıdaki sonucu onaylarım.

Prof. Dr. Hakan KARABÖRK
Enstitü Müdürü

TEZ BİLDİRİMİ

Bu tezdeki bütün bilgilerin etik davranış ve akademik kurallar çerçevesinde elde edildiğini ve tez yazım kurallarına uygun olarak hazırlanan bu çalışmada bana ait olmayan her türlü ifade ve bilginin kaynağına eksiksiz atıf yapıldığını bildiririm.

DECLARATION PAGE

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Muhammed Esat DERE
Tarih: 01.07.2019

ÖZET

YÜKSEK LİSANS TEZİ

MOBİL ROBOTLAR İÇİN OPTİMUM YOL BULMA

Muhammed Esat DERE

Konya Teknik Üniversitesi
Lisansüstü Eğitim Enstitüsü
Elektrik-Elektronik Mühendisliği Anabilim Dalı

Danışman: Dr. Öğr. Üyesi Akif DURDU

Yıl 2019, 65 Sayfa

Jüri
Dr. Öğr. Üyesi Akif DURDU
Prof. Dr. Cemil SUNGUR
Dr. Öğr. Üyesi Burak YILMAZ

Bu tezde mobil robotlar için kullanılan yol planlama algoritmaları incelenmiştir. Robotlar çoğunlukla belirli tekrarlayıcı işlevleri yerine getirmek üzere programlanmış mekanik aygıtlardır. Ayrıca insanlar için karmaşık veya tehlikeli olabilecek pek çok görevi yerine getirmek için de programlanabilirler. Robotların daha etkin ve kaynakların daha verimli kullanılabilmesi için başlangıç noktaları ile hedef noktaları arasındaki mesafeyi en az maliyetle geçmeleri gerekmektedir. Bunun için çok sayıda yol planlama algoritması geliştirilmiş olup, hangi görev robotu için hangi algoritmanın verimli olduğu önem kazanmaktadır. Bu çalışmada mesafe olarak optimum sonuçları veren A* algoritması ile en hızlı sürede sonuç veren algoritmalardan olan PRM algoritması detaylı olarak incelenmiş olup, PRM algoritmasını iyileştirecek öneriler sunularak çeşitli testler yapılmıştır. Çalışmada A* algoritmasının verdiği çıktılar optimum olarak kabul edilmiş, diğer algoritma çıktıları ile buna göre kıyaslanmıştır. PRM algoritması her çalıştırıldığında farklı sonuç verdiğinden dolayı, bu algoritmayı kararlı hale getirebilmek için ortamdaki engellerin köşe noktalarına da düğüm ataması yapılmış ve farklı ortamlar için bu yöntem test edilmiştir. Uygulanan bu yeni ve özgün yöntem sayesinde PRM algoritmasının A* algoritması gibi kararlı sonuçlar verdiği görülmüştür.

Anahtar Kelimeler: A* algoritması, mobil robotlar, optimizasyon algoritmaları, PRM algoritması, yol planlama algoritmaları

ABSTRACT

MS THESIS

OPTIMUM PATH PLANNING FOR MOBILE ROBOTS

Muhammed Esat DERE

Konya Technical University
Institute of Graduate Studies
Department of Electrical-Electronic Engineering

Advisor: Asst. Prof. Dr. Akif DURDU

Year 2019, 65 Pages

Jury
Asst. Prof. Dr. Akif DURDU
Prof. Dr. Cemil SUNGUR
Asst. Prof. Dr. Burak YILMAZ

In this thesis, the path planning algorithms used for mobile robots were investigated. Robots are mostly mechanical devices programmed to perform certain repetitive functions. They can also be programmed to perform many tasks that can be complex or dangerous for people. In order to use robots more efficiently and to use resources more efficiently, they must pass the distance between starting points and target points with minimum cost. Many path planning algorithms have been developed for this purpose and it is important to know which algorithm is efficient for task robot. In this study with A* algorithm that giving optimum result and PRM algorithm which is one of the fastest performing algorithms have been examined in detail, recommendations to improve the PRM algorithm were presented and various tests were performed. In the study the outputs given by A* algorithm are accepted as the shortest way, other algorithm outputs are compared accordingly. Therefore to enable this algorithm to be stabilized, a node assignment has been made to the corner points of the obstacles in the environment and this method has been tested for different environments. With this new and unique method it has been observed that PRM algorithm gives stable results such as A* algorithm.

Keywords: A* algorithm, mobile robots, optimization algorithm, PRM algorithm, path planning

ÖNSÖZ

Bu çalışmanın yürütülmesinde beni bilgi ve tecrübesiyle yönlendiren, her konuda anlayış gösteren ve gerektiğinde gösterdiği yolda benimle beraber yürüyen danışman hocam Dr. Akif DURDU'ya;

Çalışmalarım sırasında desteklerini ve yardımlarını esirgemeyen Konya Teknik Üniversitesi Mühendislik Fakültesi Elektrik-Elektronik Mühendisliği Robotik Otomasyon ve Kontrol Laboratuvarı RACLAB'a;

Bugünlere gelmemde büyük pay sahibi olan aileme;

Yoğun çalışmalarım nedeniyle gösterdikleri ilgi, anlayış ve bana verdikleri destekler için eşim Emine DERE'ye ve kızım Bahar DERE'ye teşekkür ederim.

Muhammed Esat DERE
KONYA-2019

İÇİNDEKİLER

KISALTMALAR

Kısaltmalar

A*	: A Star, A Yıldız.
PRM	: Probabilistic Roadmap, Olasılıklı Yol Haritası.
RSR	: Rectangular Symmetry Reduction, Dikdörtgen Simetri Azaltma.
JPS	: Jump Point Search, Atlama Noktası Arama.
GA	: Genetic Algorithm, Genetik Algoritma.
RRT	: Rapidly Exploring Random Tree, Hızlı Keşfeden Rastgele Ağaçlar.
VD	: Voronoi Diagram, Voronoi Diyagramı.
VG	: Virtual Graphic, Görünürlük Grafiği.
İHA	: İnsansız Hava Aracı, Unmanned Aerial Vehicle.
SBP	: Sample Based Path Planning, Örnekleme Tabanlı Yol Planlama.
CPU	: Central Processing Unit, Merkezi İşlem Birimi.
DPPA	: Dynamic Path Planning Algorithm, Dinamik Yol Planlama Algoritması.
PSO	: Particular Swarm Optimization, Parçacık Sürü Optimizasyonu.

1. GİRİŞ

"Robot" sözcüğü, ünlü Hollywood insan karakterlerinin imgelerini uyandırır; ancak robotlar, çoğunlukla belirli tekrarlayıcı işlevleri yerine getirmek üzere programlanmış çekingen olmayan mekanik aygıtlardır. Sürekli tekrarlanan, kirli veya tehlikeli olduğundan insanların yapmak istemediği pek çok görevi yerine getirmek üzere kullanılırlar. Ayrıca insanlar için çok karmaşık bazı görevleri yerine getirmek üzere programlanabilirler. Robotlar ekseriyetle endüstriyel olarak sınıflandırılır; oto montaj hatlarında veya servislerde kaynak yapanlar gibi. Bu günlük yaşamı en açık şekilde etkiledikleri hizmet kategorisidir. Ancak robotlar teknolojik açıdan daha ileri ve özerk hale geldikçe, işleri insanlardan daha iyi ve hızlı yapmayı öğrendiler. Bu da insanları robotlara bağımlı hale getirmeye ve robotlar olmadan yapılamayacak işler doğurmaya başlamıştır.

Robotikte bir robotun bir noktadan bir başka noktaya nasıl taşınacağı ile ilgili problem yol planlaması olarak tanımlanır. Yol planlaması belirsiz ortamlar, multirobotlar veya dinamik ortamlar gibi birçok komplikasyon da içermektedir.

Genel olarak robotikte yol planlama basit veya daha karmaşık geometrik modelleri işleyerek yararlı hareketler üreten algoritmalar tasarlamaya odaklanır. Robotikte yol planlama üç ana gruba ayrılmaktadır; hareket planlama, yörünge planlama ve belirsizlik altında planlama. Hareket planlama ve yörünge planlaması, algoritma oluşturma aşamasında temel ihtiyaçlar olarak tanımlanmaktadır. Bu durum Piyano Taşıyıcısı Problemi (Piano Mover’s Problem) ile açıklanabilmektedir. Algoritmaya girdi olarak bir evin planı ve bir piyano tanımlanır. Algoritma, piyanoyu bir odadan diğerine hiçbir şeye çarpmadan nasıl hareket ettireceğinizi belirlemelidir. Robotun yol planlaması da benzer şekilde tanımlanmıştır. Bununla birlikte hareket planlama genellikle dinamikleri ve diğer kısıtlamaları görmezden gelir ve öncelikle kontrol edilen nesne - robotun çevirilerine ve rotasyonlarına odaklanır. Bu alandaki son araştırmalar, belirsizlikler, farklı kısıtlamalar, eniyileme vb. gibi diğer hususları da dikkate almaktadır. Yörünge planlaması genellikle çözümü bir hareket planlamasından alma ve bu çözümün robotun mekanik sınırlamaları bakımından nasıl ilerleyeceğini belirleme problemini ifade eder.

Uygulamada mobil robotlar kapalı bir ortamda gezinerek belirli bir görevi yerine getirirler. Bir robottan ortamın haritasını oluşturması, haritadaki yerini kesin olarak belirlemesi veya engellere çarpmadan belirli bir yere varması istenmektedir. İlk iki görev SLAM (Eşzamanlı Yerelleştirme ve Haritalama) adı verilen bir konuya bağlıdır.

Bu konu iyi bilinen bir problemdir ve günümüzde bu probleme yönelik birçok açık kaynak çözüm bulunmaktadır (örn. HectorSLAM, OpenSLAM vb.). Robotun bir ortamda gezinmesi, ortamı bilmesine bağlıdır. Robotun herhangi bir ortam bilgisi yoksa, bu ortamı "tanıması" için sensörlerinden gelen veriye ve bunların değerlendirmesine ihtiyaç duyar. Böylece bir konumdan diğerine gidebilir. Robotun sadece sensörlerinden gelen uyarılara göre tepki verdiği yaklaşıma reaktif navigasyon denir. Bununla birlikte, robot çevre üzerinde kesin bir haritaya sahipse, optimal şekilde gezinmek için birçok tekniği kullanabilir. Bu teknikler yol planlama veya genel navigasyon olarak tanımlanmaktadır.

Her yol planlama yöntemi bir durum-uzay modeline bağlıdır. Durum-uzay modeli bir robotun tüm olası pozisyonlarını ve yönlerini temsil etmektedir. Robotik alandaki durum-uzay modelinin en çok kullanılan gösterimi grid (metrik) haritasıdır. Geometrik veya topolojik gösterimlerinde bulunmasına rağmen, grid haritalamanın ortam hakkında yeni bilgileri güncellemesi kolaydır. Bununla birlikte, grid haritasındaki her bir hücre az miktarda bir alanı temsil eder. Bu nedenle metrik harita büyük boyutlara sahip olabilmektedir. Bu tezde değerlendirilen algoritmalar ortamın grid gösterimini temel almaktadır.

Tezde, bilinen ortamlarda en kısa yolu bulmak için sık sık kullanılan A* algoritması ve PRM (Olasılıklı Yol Haritası) algoritması incelenmektedir ancak karşılaştırma yapabilmek adına Genetik Algoritma ve RRT (Hızlı Keşfeden Rastgele Ağaçlar) algoritması da tanıtılmıştır. Belirlenen haritalar için en kısa yolu bulmakta başarısını kanıtlayan A* algoritması, sonuç üretmek için uzun sürelere ihtiyaç duymaktadır. Bu da gerçekte çalıştırılacak bir robot için kullanışlı olmamaktadır. Bunun için çok hızlı sonuç veren PRM algoritması kullanılmıştır. Ancak PRM algoritması mevcut haliyle kararlı sonuç vermemektedir. Yapılan çalışmada önerilen yöntemlerle PRM algoritması kararlı hale getirilmeye çalışılmıştır ve sonuçlar test edilmiştir.

Şekil 1.1. Yol Planlama Algoritmaları

Şekil 1.1'e göre sınıflandıracak olursak A* algoritması sezgisel tabanlı bir algoritmadır. Genetik Algoritma, Parçacık Sürü Optimizasyonu algoritması da doğadan ilham alan algoritmalar sınıfındadır. PRM ve RRT ise klasik algoritmalar sınıfından örnekleme tabanlı algoritmalardır.

2. KAYNAK ARAŞTIRMASI

Yol planlayıcıları global ve yerel olarak iki başlıkta incelenmektedir. Global yol planlayıcısı genellikle bilinen bir çevre haritasına veya çevrenin mevcut ve geçmiş algısal bilgilerine dayanarak düşük çözünürlüklü bir üst düzey yol oluşturur. Bu yöntem, optimize edilmiş bir yol üretmek için değerlidir ancak bilinmeyen veya dinamik engellere tepki vermekte yetersizdir. Öte yandan yerel yol planlayıcısı, çevre hakkında önceden bir bilgiye ihtiyaç duymaz. Genellikle, yerleşik sensörlerden gelen bilgilere dayanarak, yalnızca global yolun bir parçası üzerinde yüksek çözünürlüklü düşük seviyeli bir yol sunar. Dinamik ortamlarda etkili çalışır. Hedef uzak mesafedeyken veya çevre dağınık olduğunda yöntem verimsizdir. Normal olarak her iki yöntemin hibrit olarak çalıştırılması verimi artırmakta ve bazı zayıflıkları ortadan kaldırmaktadır (Bi, Yimin, & Wei, 2008; Zhang, Butzke, & Likhachev, 2012).

Örneklemeye dayalı yol planlama algoritmaları (SBP) karmaşık veya zaman açısından kritik gerçek dünya planlama problemlerinde yeteneklerinden dolayı büyük dikkat çekmiştir. Muhtemelen, bugüne kadarki en ikna edici SBP'ler olasılıklı yol haritası (PRM) ve hızlı keşfeden rastgele ağaçları (RRT) içermektedir (Lee, Kwon, Zhang, & Yoon, 2014). Bağlantı noktalarının rastgele örneklenmesi fikri her iki yaklaşımda da temel olmakla birlikte bu iki yöntem, noktaları birleştirme tarzları bakımından farklıdır (Karaman & Frazzoli, 2011).

PRM algoritması, yüksek boyutlu durum uzayda iyi sonuçlar vermiştir. PRM, robotun boş alanda herhangi bir yere gitmesini sağlayan eğriler veya düz çizgiler ile oluşturulur. İyi bilinen iki PRM yöntemi olan Görünürlük Grafiği (VG) ve Voronoi Diyagramı (VD) çok farklı yol türleriyle çok iyi sonuçlar elde etmişlerdir. VG, engellere mümkün olduğu kadar yaklaşan bir grafiktir. Bu yöntem uygulanarak en kısa yol bulunur; bununla birlikte yol köşelerdeki veya kenarlardaki engellere dokunur, bu nedenle robot için tehlikelidir. Bunun aksine VD robot ile engeller arasındaki mesafeyi maksimize etme eğiliminde bir yol yaratır. Bu nedenle Voronoi diyagramına dayanan çözüm yolları, yol uzunluğu açısından optimum değildir. Bu yöntemin avantajı robot navigasyon görevinde yalnızca sınırlı sayıda sensör kullanılmasıdır. (Nazif, Davoodi, & Pasquier, 2010; Yan, Jouandeau, & Cherif, 2013) 'de PRM tekniği kullanılarak bir robot kümesinin yol planlaması önerilmiştir. (Hsu, Latombe, & Kurniawati, 2006; Ladd & Kavraki, 2004) tarafından çeşitli iyileştirmeler önerilmiştir.

RRT yüksek boyutlu durum-uzayda hesaplama verimliliği, yararlılığı ve nispeten hızlı şekilde uygulanabilir bir hareket planı bulma kabiliyetinden dolayı dikkat

çekmiştir. RRT büyüyen bir ağaç inşa eder. Rastgele noktalar oluşturarak yeni dallar yaratır ve bunları engelsiz bir yolun elde edildiği en yakın noktaya bağlayarak robot çalışma alanını araştırır. RRT'deki bir problem randomize tekniği kullanılarak ve birçok dal ile yol üretilerek çözülmektedir.

PRM Algoritması:

PRM yaklaşımı yol haritası oluşturma maliyetini en aza indirir ancak özellikle yüksek boyutlu örnekler için yol haritası, hızlı çevrimiçi sorgulama için çok büyük olabilir. J. Marble ve arkadaşları yaptıkları çalışmada PRM ile Grafik Teorisini birleştiren farklı alternatifler önermektedirler (Marble & Bekris, 2013). İlk alternatif sıralı yaklaşımdır; PRM'den elde edilen yol haritasına grafik anahtar algoritması (graph spanner algorithm) uygulanır. İkinci alternatif ise yol haritası için gerekli olmayan kenarların dikkate alınmamasıdır. Bu çalışmada önerilen temel kavram, asimptotik yakın-optimalliktir (daha fazla yineleme yapıldıkça çözümlerin optimale yaklaşmaları garanti edilmektedir). Önerilen iki algoritma ile k-PRM'in farklı ortamlardaki sonuçları üzerine karşılaştırma yapılmıştır. Bir zaman tasarrufunun elde edilip edilmediği, engellerin yoğunluğu ve çarpışma kontrolünün ne kadar maliyetli olduğu çevrenin özelliklerine bağlıdır. Bu çalışma, asimptotik yakın-optimalliği garanti eden konfigürasyon alanlarında seyrek yol haritalarının hesaplanmasının pratik olduğunu göstermiştir. Bu yol haritaları asimptotik olarak en uygun yollara sahip yol haritalarından çok daha az kenarlara sahipken, yol kalitesinde küçük bozulmalara neden olmaktadır. Germe faktörü parametresi bu değişimin denge ayarını yapma olanağı sağlar. Deneyler, düşük gerilimli faktörlere sahip yol haritalarının yüksek yol kalitesine sahip olduklarını ancak daha yoğun olduklarını göstermektedir. Mevcut yaklaşım sadece yol haritası kenarlarını kaldırır.

R. Pearce ve arkadaşları, yol haritası yapısının iyileştirilebileceği örnekleri seçerek, yol haritası yapım performansını iyileştirmek amacıyla PRM için bir filtreleme stratejisi sunmaktadır (Pearce, Morales, & Amato, 2009). Mevcut yol haritasına göre örneklemenin yapısal iyileştirmesi ölçülmekte, sadece iyileştirme için yeterli potansiyele sahip örnekler seçilmektedir. Düğüm üretimi rastgele örneklemeden, onların test edilmesinden ve geçerli olanların yol haritasında tutulmasından oluşmaktadır. Yapılan tüm testler akıllı örnekleme planlarının gücünü göstermektedir. Çoğu durumda düğüm seviyesinde filtreleme ile birleştirilen akıllı bir plan, önemli iyileştirmelere yol açabilmektedir.

B. Paden ve arkadaşları, PRM'in sorgu aşamasındaki çalışma süresini azaltmak

için nokta tabanlı bir sezgisel tarama araştırması yapmışlardır (Paden, Nager, & Frazzoli, 2017). Bu çalışmada PRM'nin sorgulama evresinin çalışma süresini kısaltmak için nokta tabanlı kabul edilebilir bir sezgisel bulgunun etkinliği incelenmektedir. Nokta tabanlı sezgisel tarama ilk olarak en kısa yol bulma sorgularının tek bir grafik üzerinde çözüldüğü yol ağlarındaki araç rota oluşturma problemleri için geliştirilmiştir (A. V. Goldberg & Harrelson, 2005). Teoride, eğer yeterince sayıda sorgu çözülmeli ise grafiği ön işleme koymak için harcanan herhangi bir süre, en kısa yol sorgularını çözmek için harcanan süreye kıyasla ihmal edilebilmektedir. Bu gözlem her yönlendirme problemine, grafik boyutuna göre sabit sürede cevap vermek için tüm çiftlerin en kısa yol problemini çözmeyi önermektedir. Ancak tüm çiftler için bulunmuş en kısa yol çözümlerini saklamak için gereken hafıza, büyük yol ağları için oldukça maliyetlidir. Nokta tabanlı sezgisel tarama az sayıdaki tek kaynaklı en kısa yol problemlerini çözerek ve çözümlerini belirli bir grafik için etkili bir sezgisel yapı oluşturmak için kullanarak bellek gereksinimleri ve sorgu süreleri arasında bir denge sağlamaktadır.

RRT Algoritması:

RRT yüksek boyutlu uzayda hızlıca arama yapmak için verimli bir veri yapısı ve örnekleme şeması olarak tanıtıldı (LaValle, 1998). Ancak yöntem hem cebirsel kısıtlamalara (engellerden kaynaklanan) hem de diferansiyel kısıtlamalara (holonomik olmayan ve dinamiklerden kaynaklanan) sahipti. Yöntemin ana fikri uzayın keşfedilmemiş kısımlarına doğru keşfe eğilimli olmaktır. J. Kuffner ve arkadaşları, RRT-Connect adıyla farklı kısıtlamaların olmadığı sorunlara göre uyarlanmış bir yaklaşım sunmuşlardır (Kuffner Jr & LaValle, 2000). Bu yöntem başlangıç ve hedef düğümlerinden iki RRT'yi art arda oluşturmak suretiyle çalışmaktadır. Ağaçların her biri etrafındaki alanı keşfetmekte ve basit açgözlü bir buluşsal yöntem kullanarak birbirlerine doğru ilerlemektedir. RRT-Connect planlayıcısı özellikle hiçbir diferansiyel kısıtlama içermeyen yol planlama problemleri için tasarlanmıştır. Yöntem daha uzun bir mesafe boyunca hareket etmeye çalışan sezgisel bağlantılar ve RRT'lerin hem başlangıç hem de hedeften büyümesi üzerine kurulmuştur. RRT-Connect için tekrarlanan deneylerle basit ve tutarlı davranışlar gözlenmiştir; açgözlü arama (potansiyel alan planlayıcısındaki gibi) ve düzgün keşif (PRM gibi) arasında makul bir denge vardır.

Özerk taşıtların ve robotik manipülatörlerin kontrolünde önemli bir unsur, tıkanık alanlarda yolları planlayabilme yeteneğidir. Dinamik kısıtlamalar altında uygulanabilir yörüngeleri bulma problemi aynı zamanda kinodinamik programlama olarak da ifade edilmektedir (Donald, Xavier, Canny, & Reif, 1993). Planlama

problemi, başlangıç pozisyonundan hedef bölgeye dinamik olarak uygulanabilir bir yol oluşturmaktır. Sıkışık alanlarda planlama yapmak için çözüm olarak bir alan modeli oluşturmak ve zor bölgelere daha fazla hesaplama kaynağı ayırmak önerilmektedir (Burns & Brock, 2005). Başka bir çözüm, konfigürasyon alanında özelliklere göre rastgele örneklemeye eğilimi artırmaktır (Zucker, Kuffner, & Bagnell, 2008). M.J. McCourt ve arkadaşları yaptıkları çalışmada sıkışık bir ortamda uygulanabilir yolları hızlı bir şekilde belirleme problemini ele almaktadır (McCourt, Ton, Mehta, & Curtis, 2016). Mevcut birçok yol planlama algoritması olsa da, pek çoğu yeni bilgilere dayanarak gerçek zamanlı yeniden planlama sorununu düşünmemektedir. Bu çalışmada, kök düğümden mesafeye bağlı olarak algoritmanın adım boyutunu ayarlayan standart RRT algoritması üzerinde bir varyasyon çalışılmaktadır. Bu adaptif adım-uzunluklu RRT algoritması, gerçek zamanlı uygulamalar için arzu edilen yollar üretmektedir. Bu geniş kapsamlı ham yollar hızlı bir şekilde üretilmekte ve belirli bir yola karar verirken daha fazla aday içeren dinamik bir planlayıcı sunmaktadır. Standart RRT daha yumuşak yollar oluştururken, adaptif adım-uzunluklu RRT alanı daha ayrıntılı bir şekilde kapsayan yollar üretmektedir. Adaptif algoritma başlangıç konfigürasyonunun yanında daha yumuşak yollara ve bu konumdan uzakta daha ham yollara sahip olmaktadır.

Verimlilik elde etmek için pratik hareket planlama yöntemleri genellikle bütünlük gereksinimlerini yumuşatmaktadır. PRM gibi algoritmaları içeren örnekleme tabanlı yaklaşımlar bu yönde nispeten yeni bir araştırma çizgisi oluşturmaktadır (Kavraki, Svestka, & Overmars, 1994). Çoğu örnekleme tabanlı algoritmalar olasılıkla tanımlanmıştır, diğer bir deyişle algoritmanın çözüme ulaşma olasılığı, örnek sayısı sonsuza yaklaştığında bire yakınsar. S. Karaman ve arkadaşları planın yürütme sırasındaki çevrimiçi yakınsamasını geliştirmek için RRT* algoritmasının her zamanki asimptotik optimallik özelliğini kullanmışlardır (Karaman, Walter, Perez, Frazzoli, & Teller, 2011). Deneysel sonuçlar bu önerilen RRT* uzantılarının yörünge kalitesini önemli ölçüde artırdığını göstermektedir. Her durumda algoritma iki engel arasındaki rotayı hedefe daha kısa yollar sağlayarak doğru şekilde oluşturmaktadır. Bazen RRT* robotu hedeften uzağa yönlendiren bir ilk çözüm sunabilmektedir. Robot yola devam ettikçe RRT* daha kısa bir yol bulmak için ağacın yapısını değiştirmektedir. Sonuç olarak temel RRT'ye göre daha kısa olma eğiliminde olan, döngüsel olmayan yollar üretilebilmektedir.

Temel RRT algoritmasının zayıf yönlerinden birisi yol maliyetini hesaba katmamasıdır. Bu, optimal olmaktan uzak çözümlere yol açabilmektedir. C. Urmson ve arkadaşları RRT algoritmasına yol maliyeti bilgisini de ekleyerek bir çözüm

geliştirmişlerdir (Urmson & Simmons, 2003). Algoritma, sezgisel kavramlar eklenerek geliştirilerek ağacın rastgele büyümesi yönlendirilmiş, hedefe giderken daha düşük maliyetli yollara doğru ağacın büyümesi sağlanmıştır. Bu sağlanırken RRT algoritmasının keşif eğilimi de korunmuştur. Güncellenen algoritma hRRT olarak adlandırılmıştır. Yapılan deneylerde hRRT algoritması ile üretilen yolların kalitesinde önemli bir gelişme sağlandığı görülmüştür.

Son zamanlarda çoklu-İHA'lar (İnsansız Hava Araçları) yalnızca askeri alanlarda değil, gözetleme ve kurtarma gibi sivil alanlarda da yaygın olarak kullanılmaktadır. Yol genellikle İHA'nın sırayla ziyaret etmesi gereken bir dizi yol düğümü cinsinden tanımlanmaktadır. İHA yol problemlerini çözmek için A* algoritması, karınca kolonisi optimizasyon algoritması ve RRT gibi birçok algoritma üzerinde çalışmalar yapılmıştır (Choset et al., 2005; LaValle, 2006). (K. Yang & Sukkarieh, 2008)'de, saha ortamlarında İHA navigasyonu için geliştirilmiş bir RRT algoritması uygulanmaktadır. Son zamanlarda planlama zamanını, yolun maliyetini veya hareket kısıtlamasını azaltmaya yönelik çeşitli optimal RRT yol planlama algoritmaları geliştirilmiştir (Amato & Wu, 1996). Bununla birlikte bunların birçoğu özellikle statik engeller içeren problemler için planlama problemini çoklu İHA'ların seyahat ortamlarında çözmeye çalışmaktadır (Dijkstra, 1959; Gu, Postlethwaite, & Kim, 2006; Saunders, Call, Curtis, Beard, & McLain, 2005). W. Zu ve arkadaşları, statik ve dinamik ortamlarda çalışacak İHA'lar için bir yol planlama algoritması geliştirmişlerdir (Zu et al., 2018). Kısa süre içerisinde bir grup İHA için engelli ortamlarda yol planlamak zorlu bir süreçtir. İlk olarak İHA'ların manevra kısıtlamalarını dikkate alarak iyileştirilmiş bir RRT önerilmekte ve yoldaki gereksiz düğümleri silmek için basit ve etkili bir yol budama yöntemi tasarlanmaktadır. İkinci olarak İHA'lar hareketli bir takım arkadaşı ya da aniden çıkan bir engel tespit ettiğinde çarpışmayı önlemek için düşük maliyetli bir yol oluşturan işbirlikçi bir yol planlama yöntemi geliştirilmiştir. Bir İHA yeni bir engel tespit ettiğinde, önce hızlı bir şekilde yeni bir yol bulunmakta ve daha sonra işbirlikçi yol planlama stratejisine bağlı olarak daha iyi bir yol üretilmektedir. Önerilen algoritmanın performansını göstermek için simülasyon çalışmaları sunulmaktadır. Amaç bir dizi İHA'nın belirli bir başlangıç noktasından hedef noktaya seyahat ederken, engel açısından zengin bir ortamda statik ve dinamik engellerden kaçınmasıdır. Tüm İHA'ların aynı yükseklikte uçtuğu varsayılmaktadır. Algoritma iyi performans göstermektedir. Statik, aniden çıkan ve dinamik engellerin varlığında çarpışmadan uzak yollar bulmaktadır. Algoritma oldukça kısa sürede bir yol bulmakta ve hesaplama süresi arttıkça çözümlerin kalitesi de artmaktadır. Dinamik çarpışma ihtimalinde, yol

planlayıcısı çarpışmayı çözmek için bir dizi kriteri temel alan yeni bir yol bulmaktadır. Yol planlayıcısının performansı çeşitli simülasyon çalışmaları ile doğrulanmakta ve tatmin edici bulunmaktadır.

RRT ve PRM gibi klasik yaklaşımları gerçek zamanlı hareket planlamasına uygulamak için birçok çalışma yapılmıştır (Aoude, Luders, Levine, & How, 2010; Desaraju & How, 2011; Lau, Sprunk, & Burgard, 2013; Park, Choi, & Chung, 2012). Bununla birlikte klasik yaklaşımlar optimal yolları üretememekte ve bazı yerel minimum noktalarda kilitlenme eğilimindedirler. Örneklemeye Dayalı Yol Planlama Algoritmaları (SBP)'nın yüksek boyutlu problemler için geçerli yollar sağlama yeteneği avantajlıdır. Ancak SBP'nin etkili stratejisinin çekirdeği olan rastgele örnekleme yaklaşımı fazladan manevra yapılmasına yol açmaktadır. Ayrıca bazıları birden fazla engel varlığında veya dinamik ortamlarda uygun çözümü sağlamayabilmektedir. Klasik yöntemlerin verimsizliğini önlemek için, sezgisel yaklaşımlar ve hibrit algoritmalar kullanılmaktadır.

A Algoritması:*

Temel A* algoritması için yüksek boyutlu bir haritadaki hesaplama süresi (bilgisayar ortamında) bir saatten fazla sürmektedir. Bu nedenle A* algoritması için çeşitli modifikasyonlar üretilmiştir. F. Duchon ve arkadaşları A* algoritmasının çeşitli modifikasyonlarını (Basic Theta*, Phi*) ve iyileştirmelerini (RSR, JPS) tanıttıkları bir çalışma yapmışlardır (Duchoň et al., 2014). Bu modifikasyonların ana çıkış noktası, temel A* algoritmasının çok uzun sürelerde çıktı vermesidir. Bu çalışmada bu süre düşürülmeye çalışılmış ve başarılı olunmuştur. Grid temelli algoritmalarda mevcuttaki hücreden sıradaki hücreye gitmek için bitişikteki sekiz hücreden birisi seçilmek zorundadır. Bu her zaman kullanışlı olmayabilir çünkü birbiri ile bağlı olan hücreler arasında uzun mesafelerde çok boşluk kalmakta ve bu hücreler zikzak stili ile bağlı olmamaktadır. Bu nedenle mevcut hücreden sıradaki hücreye geçmek için açı temelli aramanın yapıldığı modifikasyonlar türetilmiştir. Açı temelli arama yapan bu uzantılar Basic Theta* ve Phi*'dır. Basic Theta* hücreler arasındaki görüş alanını test eden bir A* uzantısıdır (Yap, Burch, Holte, & Schaeffer, 2011). Bu, sıradaki seçilen hücre doğrudan görüş alanı içerisinde ise aradaki hücrelerin göz ardı edilerek o hücreden devam edilmesi anlamına gelmektedir. Bu şekilde yalnızca robotun geçmesi gereken hücreler bulunmaktadır. Phi* algoritması, Basic Theta* algoritmasının bir uzantısıdır (Nash & Koenig, 2013). Bu algoritma her hücre için iki açı değeri kaydetmektedir. Phi* algoritmasının bu özelliği, robotun dinamiklerini algoritmanın hesaplanmasına dahil etmek için kısıtlayıcı bir şekilde izin vermektedir. Ancak bu algoritmaların gelişmiş

yetenekleriyle yolu gerçek zamanlı olarak planlamak mümkün olmamaktadır. Bu algoritmalar hala karmaşık grid haritaları kullanmaktadırlar. RSR (Dikdörtgen Simetri Azaltma) uygulamasının geliştirilme nedeni budur. RSR bir önişleme adımıdır ve simetrileri tanımlayarak ortadan kaldırmaktadır. RSR'nin avantajı düşük işlem talebi ve yıldız algoritmaları ailesiyle kombinasyon olasılığıdır (D. Harabor, 2012). Simetriler, içerisinde sadece boş alanın oluştuğu dikdörtgenler olarak tanımlanmaktadır. Bu dikdörtgenlerin içindeki yol yalnızca dikdörtgenin kenarları tarafından belirlenmektedir. İncelenen hücre miktarının azaltılmasının bir başka yöntemi, doğrudan A* algoritmasına uygulanabilen JPS (Atlama Noktası Arama) yöntemidir (D. D. Harabor & Grastien, 2011). Bu yöntemin ilkesi, değerlendirilmiş olan hücrenin çevresindeki komşu hücrelerin kırpılmasıdır. Süre açısından en iyi sonucu veren JPS algoritması olmuştur. Dezavantajı ise bazen temel A* algoritmasının bulduğu sonuçtan daha uzun bir yol bulmasıdır. Bu nedenle sürenin önemli olduğu durumlarda tercih edilebilmektedir. Yolun uzunluğu önemli ise Basic Theta* algoritması kullanılabilir.

Bilgisayar oyunları, küçük ortamlardaki yol planlama problemleri ve diğer birçok alan için haritalar çok büyük olmadığından A* gibi geleneksel algoritmalar iyi çalışmaktadır. Ancak büyük bir şehirde veya ülkede iki nokta arasındaki en kısa yolun bulunması isteniyorsa sınırlı bellek ve CPU kaynakları ile geleneksel algoritmalar kötü performans sergileyebilmektedir. H. Wang ve arkadaşları büyük haritalarda gezinebilmek için geleneksel A* algoritmasına hiyerarşik mekanizmayı ekleyen, CPU ve bellek kaynaklarını koruyan Hiyerarşik A* algoritmasını önermişlerdir (Wang, Zhou, Zheng, & Liang, 2014). Ülke haritası gibi büyük bir haritada iki nokta arasındaki en kısa yolu bulmak için ülke haritası sırası ile bölgelere, şehirlere ve mahallelere bölünmektedir. Sonra her katman için A* algoritması çalıştırılmakta ve yinelemeli bir şekilde bu iki nokta arasındaki en kısa yol bulunmaktadır. A* algoritmasının standart formülünde bulunan sezgisel fonksiyon h(x), dünya yuvarlak olduğu için düz bir çizgi olarak değil, başlangıç ve bitiş noktalarının enlem ve boylamı da dahil edilerek hesaplanmaktadır. Uzun mesafelerde Hiyerarşik A* algoritmasının temel A* algoritmasına göre daha iyi sonuç verdiği görülmektedir.

Xiang Liu ve arkadaşları labirent gibi ortamlarda arama kurtarma yapacak olan robotlar için A star algoritmasını kullanmışlardır (Liu & Gong, 2011). Çalışmada üç farklı A star algoritması denklemi oluşturulmakta, her denklemde farklı sezgisel fonksiyon h(x) kullanılmaktadır. Bu üç algoritma ile sezgisel fonksiyona sahip olmayan Derinlik Öncelikli (Depth-First) yol bulma algoritması on adet labirent için çalıştırılmaktadır. Deney sonuçlarına göre A* algoritmaları birçok labirentte Depth-First

algoritmasından daha iyi sonuç vermektedir. Bu sonuç sezgisel bilginin labirent aramalarında çok yardımcı olduğunu ve sezgisel bilgiye sahip bir algoritmanın sezgisel bilgiye sahip olmayan bir algoritmadan daha iyi performans gösterdiğini doğrulamaktadır. Ayrıca açıya dayalı sezgisel fonksiyonun bu çalışmada iyi sonuç vermediği görülmüştür. Neden olarak da açı değerine sapmalar eklenmesi öne sürülmüştür.

Genetik Algoritma:

Genetik algoritmalar (GA), Darwin'in evrim hakkındaki teorisinden ilham alan doğal seleksiyon ilkelerini kullanan arama algoritmaları ve optimizasyon teknikleridir (en güçlülerin hayatta kalması). GA tabanlı yaklaşımlarda değişkenler bir kromozomdaki genler olarak temsil edilmektedir. Robot yolu planlama alanında GA'ya dayalı çeşitli çalışmalar yapılmıştır. (Hu & Yang, 2004)'de bu konu standart GA yerine, bilgiye dayalı bir genetik algoritma (probleme özgü genetik algoritma) uygulanarak çözülmektedir. Algoritma hem alan bilgisi hem de küçük ölçekli yerel arama ile tasarlanmıştır. Önerilen yöntem hem statik hem de dinamik ortamlar için uygundur. Bu algoritma dinamik ortamlardaki birden fazla mobil robot için de genişletilmiştir (S. X. Yang, Hu, & Meng, 2006). Robot çalışma alanı basit olduğunda veya çok hızlı değişmediğinde, bu yaklaşım mevcut konumdan hedefe en yakın optimum çözümü sunmaktadır. Çok karmaşık, hızlı ve dinamik bir ortamda, uygulanabilir bir yol her zaman üretilmektedir.

Mobil robot dar açılı "U" veya "V" şeklinde bir engelle veya bir dinamik engelle karşılaştığında başlangıç noktasından hedef noktasına çarpışmasız bir yol oluşturmak için GA tabanlı dinamik yol planlama algoritması (DPPA) önerilmiştir (Yun, Parasuraman, & Ganapathy, 2011). Ek olarak bu algoritma hedefe yöneliktir ve böylece arama süresini azaltmaktadır.

(Oleiwi, Al-Jarrah, Roth, & Kazem, 2014)'de B.K Oleiwi ve arkadaşları GA, modifiye edilmiş A* algoritması ve bulanık mantığa dayalı bir hibrit yaklaşım sunmaktadır. A* algoritması yol haritası yaklaşımında veya düzenli bir gridde olduğu gibi rota maliyetini en aza indirgeme eğilimindedir. Her düğümü veya hücreyi, robotun başlangıç ve hedef konumları arasındaki mesafesine göre değerlendirmektedir. İlk olarak, modifiye GA ve modifiye A* algoritması en uygun yolu bulmak için uygulanmaktadır. Daha sonra global optimum yörünge, zamana dayalı bir yörüngeyi yeniden oluşturmak için bulanık hareket kontrolörüne girdi olarak kabul edilmektedir. Yörüngede bilinmeyen engeller göründüğünde, bulanık denetleyici robotun hızını

azaltmaktadır. Algoritma dinamik engelli ortamlarda çok etkileyici sonuç vermektedir.

Hareketli hedefi izlemek için bulanık mantık ve GA'ya dayalı bir yöntem sunulmaktadır (Benbouabdallah & Qi-dan, 2013). Hibrit meta-sezgisel GA – PSO algoritması uygulanarak optimal otonom robot yolu elde edilmektedir (Huang & Tsai, 2011). GA ve PSO kombinasyonu çaprazlama ve mutasyon operatörleri çalıştırılarak yeni bir çözüm önerilmektedir. Sonuç olarak geleneksel GA ve PSO algoritmasındaki erken yakınsamadan kaçınılmaktadır.

J. Tu ve arkadaşlarının önerdiği algoritmada, başlangıç noktasından hedefe giden herhangi bir yol, neslin ürettiği bir çözümdür (Tu & Yang, 2003). İlk nesil rulet tekerleği yöntemi ile seçilmekte, yani başlangıçta büyük bir rastgele dizi popülasyonu üretilmektedir. Kabul edilemez çözümleri temsil eden diziler ortadan kaldırılmakta ve kabul edilebilir çözümleri temsil eden diziler çoğaltılmaktadır. Kabul edilemez çözümler, hedefe ulaşamayan dizilerdir. Kabul edilebilir çözümler, hedefe ulaşabilecek dizilerdir. Mobil robotu hedefe en kısa yoldan ulaştıracak en iyi dizi bulunmalıdır. Bu kararlara dayanarak popülasyondaki her diziye bir uygunluk değeri atanmaktadr. Kabul edilebilir çözümler daha yüksek uygunluk değerine sahip olacak ve kabul edilemez bir çözüm daha düşük uygunluk değerine sahip olacaktır. Bu nedenle, daha yüksek uygunluk değerine sahip bir dizi, mobil robotun hedefe doğru hareket ettiğini göstermektedir. Algoritma hem statik hem de dinamik ortamlarda bir mobil robot için kullanılabilmektedir. Statik ortamlar için, üretilen robot yolu matematiksel bir görünümle en uygun duruma getirilmektedir. Dinamik ortamlar için üretilen robot yolu ise en uygun değerlere yakındır.

Son yirmi yılda hücre ayrışması, yol haritası ve potansiyel alan gibi yol planlama problemlerini çözmek için farklı geleneksel yöntemler geliştirilmiştir (Barraquand & Latombe, 1991; Graf, Wandosell, & Schaeffer, 2001). Bu yöntemlerin çoğu, alan konfigürasyonu kavramına dayanmaktadır (Lozano-Pérez & Wesley, 1979). Bu teknikler uyum eksikliği ve sağlam olmayan bir davranış göstermektedir. Bu yaklaşımların zayıflığının üstesinden gelmek için araştırmacılar çeşitli çalışmalar yapmışlardır (Elshamli, Abdullah, & Areibi, 2004). GA, karmaşık ve kötü niyetli optimizasyon problemleri için en sağlam arama algoritmalarından biri olarak kabul edilmektedir (D. E. Goldberg, 1989). GA'yı bu tür problemleri çözme konusunda çekici kılan temel karakteristik özelliği, doğal olarak paralel arama teknikleri olması ve dinamik ortamlarda optimal arama yapabilmeleridir (Branke, 2003; Thomaz, Pacheco, & Vellasco, 1999). (Lu & Yang, 2007)'de önerilen tekniklerden bazıları birçok soruna

göz yummuştur. Bunlar, (1) maliyetlidir, (2) dinamik ve büyük boyutlu ortamlarla uğraşırken büyük bellek alanları gerektirir, (3) zaman alıcıdır. (Jung, Hong, Hong, & Hong, 1999; Ramirez-Serrano & Boumedine, 1996)'da yazarlar yol planlama problemini ele almak için sırasıyla Bulanık Mantık ve Yapay Sinir Ağı'nı kullanmışlardır.

İ. AL-Taharwa ve arkadaşları yaptıkları çalışmada, kontrol edilebilir bir mobil robotun gridlere ayrılmış ortamda bir başlangıç ve bitiş noktası arasındaki optimum yolu bulmasına yardımcı olmak için genetik algoritmaları kullanmışlardır (Ismail, Sheta, & Al-Weshah, 2008). Mobil robot, başlangıç noktası ile bitiş noktası arasındaki mesafeyi azaltan en uygun yolu bulmak zorundadır. GA'lar, indirgeme tabanlı yöntemler gibi geleneksel arama tekniklerinin karşılaştığı birçok sorunun üstesinden gelebilmektedir. Önerilen kontrol algoritması dört komşu harekete izin vermekte, böylece yol planlaması düşük karmaşıklığa sahip arama alanlarına uyum sağlayabilmektedir. Ön deneyler, önerilen yaklaşımın statik ortamlardaki farklı görev türlerini yerine getirmede etkili ve verimli olduğunu göstermektedir.

GA 1975 yılında ortaya çıkmasından günümüze kadar birçok optimizasyon problemini başarıyla çözmüştür (Holland, 1975). Y. Hu ve arkadaşları grid'leri birleştiren ve temsilleri koordine eden basit ama etkili bir yol gösteriminin kullanıldığı bilgi tabanlı bir GA önermişlerdir (Hu & Yang, 2004). Diğer grid yöntemlerinden farklı olarak burada benimsenen grid, yolun hareketini sınırlamamakta fakat çevreyi ayrıştırıp kromozom yapısını ve genetik operasyonu basitleştirmektedir. Bu yaklaşım her gen için bir sayıya sahip olmayı ve kromozomlarda gerçek sayılar yerine tamsayıları kullanmayı mümkün kılmaktadır. Probleme özgü genetik operatörler sadece alan bilgisi ile tasarlanmamıştır, aynı zamanda operatörlerin verimliliğini artıran küçük ölçekli yerel aramaları da içermektedir. Hem uygulanabilir hem de uygulanabilir olmayan çözümlere nispeten basit ama etkili bir değerlendirme yöntemi uygulanmaktadır. Algoritma hem statik hem de dinamik ortamlar için uygundur. Yapılan çalışmada GA aynı zamanda etkin değerlendirme yöntemine sahiptir. Bu, uygulanabilir çözümlerden daha iyi çözümler geliştirmek için oldukça faydalıdır. Bilgi tabanlı operatörlerin etkinliği simülasyon çalışmaları ile gösterilmiştir. Simülasyon sonuçları ayrıca önerilen genetik algoritmanın hem karmaşık statik hem de dinamik ortamlarda etkili olduğunu göstermektedir.

Ticari havayolları sürekli öngörülen yörüngelerde uçarken, operasyonel bölgelerdeki İnsansız Hava Araçları'nın (İHA) uçuş sırasında geçerli olan belirli alan ve

koşullara bağlı olarak sürekli değişen yörüngeleri gezmeleri gerekmektedir. Geçmişte, en iyi yol en kısa yolla ilişkilendirilmiş ve en kısa yolu bulmak için deterministik arama algoritmaları kullanılmıştır. Sorunun tanımı o zamandan beri evrim geçirmiştir ve en iyi yol gidilen mesafeyi, ortalama rakımı, yakıt tüketimini, radar maruziyetini vb. en aza indiren yolla ilişkilidir. Bunlar, dikkate alınması gereken faktörlerin birkaç örneğidir ve sorunun karmaşıklığının arttığını açıkça göstermektedir. Bu karmaşıklıkla başa çıkabilmek için araştırmacılar deterministik algoritmalar (matematikte, bilgisayar bilimlerinde ve fizikte deterministik sistem, sistemin gelecekteki durumlarının geliştirilmesinde rastlantısallığın bulunmadığı sistemlerdir.) kullanmaktan, deterministik olmayan algoritmalar kullanmaya geçtiler (Masehian & Sedighizadeh, 2007). V. Roberge ve arkadaşları yaptıkları çalışmada, Genetik algoritmayı (GA) ve parçacık sürüsü optimizasyon algoritmasını (PSO) aracın dinamik özelliklerini göz önünde bulundurarak, problemin karmaşıklığı ile başa çıkarmak ve karmaşık bir 3B ortamdaki sabit kanatlı İHA'lar için uygun yörüngeleri hesaplamak için kullanmışlardır (Roberge, Tarbouchi, & Labonté, 2012). Optimal yolun özellikleri, geliştirilen çok amaçlı bir maliyet fonksiyonu biçiminde gösterilmiştir. Üretilen yollar; çizgi parçalarından, dairesel yaylardan ve dikey spirallerden oluşmaktadır. “Tek program, çoklu veri” paralel programlama paradigmasını kullanarak çözümlerin uygulama süresi kısaltılmış ve standart ticari kullanıma hazır çok çekirdekli işlemcilerde gerçek zamanlı performans elde edilmiştir. Çalışmada GA ile Parçacık Sürü Optimizasyonu (PSO) karşılaştırılmıştır. PSO, 1995 yılında Kennedy ve Eberhart tarafından önerilen, popülasyon temelli, deterministik olmayan bir optimizasyon yöntemidir (Kennedy & Eberhart, 1995). GA'ya benzer şekilde, PSO son zamanlarda sonsuz dürtü yanıtlı dijital filtrelerin tasarımı ve optimizasyonu gibi çok çeşitli uygulamalarda kullanılmıştır (Slowik & Bialko, 2007). Algoritma, bir parçacık sürüsünün, çok boyutlu bir arama alanındaki optimal bir çözüme doğru ilerleyen hareketini simüle etmektedir. Her parçacığın konumu bir aday çözümü temsil etmekte ve rastgele başlatılmaktadır. Uygulamada, bir parçacığın konumu GA ile aynı kodlamayı kullanan tam bir yörüngeyi temsil etmektedir. Yinelemeli işlemin her aşamasında, her parçacığın hızı, parçacığın önceki hızına, parçacık tarafından işgal edilen en iyi pozisyona (kişisel etki) ve kümenin herhangi bir parçacığının işgal ettiği en iyi pozisyona dayalı olarak ayrı ayrı güncellenmektedir (sosyal etki). Deney sonuçlarına dayanılarak çok çekirdekli ticari kullanıma hazır işlemcilerde 10 saniyelik sabit bir hesaplama süresinde İHA'lar için yol planlama problemini çözerken GA'nın PSO'ya tercih edilebileceği sonucuna varılmıştır.

GA bir grid haritasında çalıştığından ve nüfus çeşitliliğini kontrol etmediğinden,

erken bir yakınsama olabilir. Bu nedenle GA robot yolu planlama uygulamalarında daha iyi sonuçlar elde etmek için A*, bulanık mantık, PSO vb. gibi diğer algoritmalarla entegre edilmesi gerekebilmektedir.

Tezde yapılan çalışmalarda örnekleme tabanlı PRM algoritması kullanılmış, PRM algoritmasını kararlı hale getirmek için özgün yöntemler önerilmiştir. Sonuçlar, sezgisel tabanlı A* algoritmasının çıktıları ile karşılaştırılarak yorumlanmıştır. Ayrıca bu iki algoritmanın çıktıları, RRT ve GA algoritmalarının çıktıları ile de karşılaştırılmıştır.

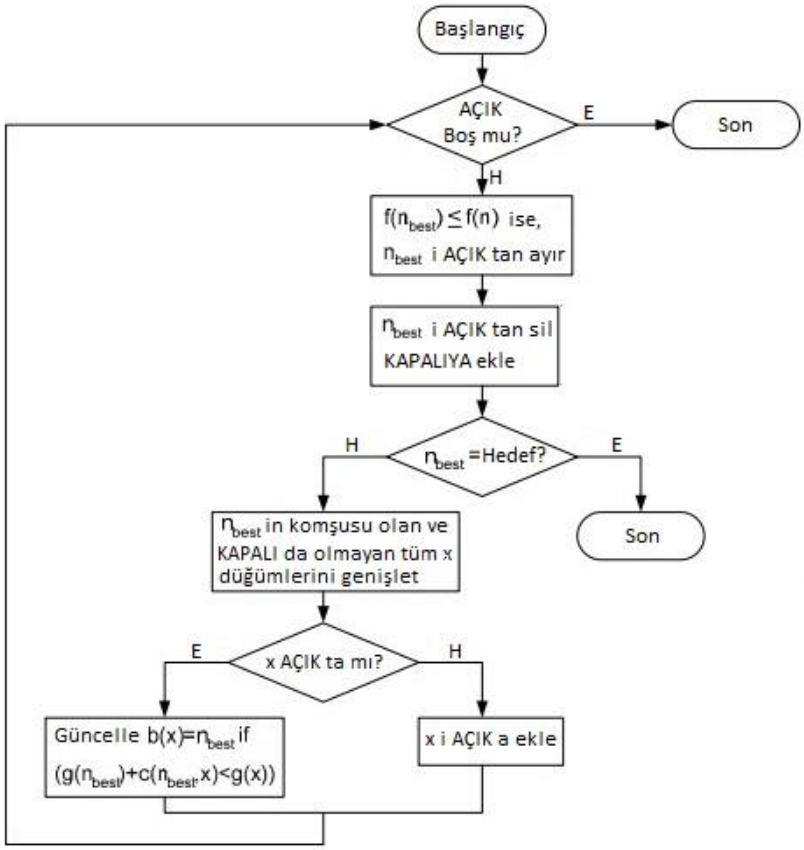
3. MATERYAL VE YÖNTEM

Tezin bu bölümünde A* algoritması, PRM algoritması, GA ve RRT algoritması tanıtılmıştır. Örnek bir ortam için elde edilen çıktılar verilmiştir.

3.1. A* Algoritması

Bilgisayar bilimlerinde A* ("A star" olarak telaffuz edilir) yol bulma alanında yaygın olarak kullanılmaktadır. Düğüm olarak adlandırılan çoklu noktalar arasındaki en verimli yolları belirleyen bir bilgisayar algoritmasıdır. Performansı ve doğruluğu nedeniyle yaygın şekilde kullanılmaktadır.

Stanford Araştırma Enstitüsü'nden Peter Hart, Nils Nilsson ve Bertram Raphael (şimdi SRI International) algoritmayı 1968'de açıklamışlardır (Hart, Nilsson, & Raphael, 1968). Edsger Dijkstra'nın 1959'da yayınladığı algoritmasının bir uzantısıdır. A*, araştırmalarını yönlendirmek için sezgisel yöntemler kullanarak daha iyi bir performans elde etmektedir.



Şekil 3.1. A* Algoritması

Şekil 3.1'de A* algoritmasının yapısı sunulmuştur. A* algoritması, metrik veya topolojik konfigürasyon alanına uygulanabilen, bilinen en iyi yol planlama algoritmalarından biridir (Cui, Wang, & Yang, 2012). Bu algoritma, en kısa yola dayalı sezgisel aramayı kullanır. A* algoritması En İyi İlk Arama algoritması (best-first) olarak tanımlanır, çünkü konfigürasyon alanındaki her bir hücre kendi değerini almaktadır:

$$f(n) = g(n) + h(n) \tag{3.1}$$

denklemindeki;

f(n) = Algoritma sezgisel fonksiyonu,

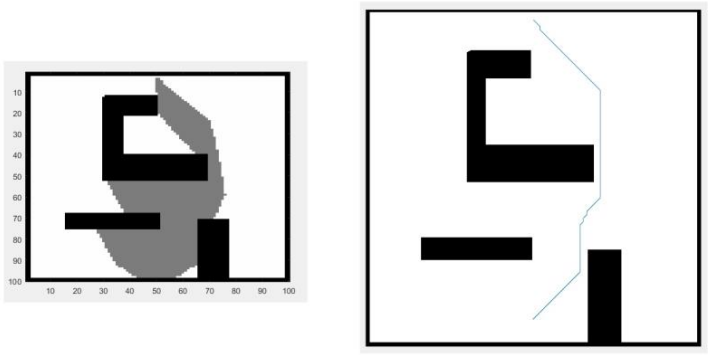
g(n) = Çıkış noktasından bulunulan noktaya ulaşma maliyeti,

h(n) = Bulunulan noktadan hedef noktaya ulaşmak için tahmin edilen uzaklık (manhattan, öklidyen veya çebişev bağıntıları ile hesaplanabilir),

olarak tanımlanmaktadır. Görüleceği gibi f(n) fonksiyonunu sezgisel yapan, bu fonksiyondaki h(n) sezgisel fonksiyonudur. Algoritma Denklem 3.1'deki toplama

işlemini kullanır. Algoritmada en öncelikli olan nokta, f(n) değeri en düşük olan noktadır. A* algoritmasının çalışması aşağıdaki gibidir:

1. Algoritma her adımda en düşük değerli noktayı alır ve o noktayı sıradan çıkarır.
2. Ulaşılan bu noktaya göre komşu olan bütün noktaların değeri yenilenir.
3. Algoritma üstteki adımları hedefe ulaşana kadar veya sırada başka nokta kalmayıncaya kadar tekrarlar.



Şekil 3.2. A* algoritmasının örnek çıktısı

Şekil 3.2'de A* algoritmasının örnek bir ortam için verdiği çıktı görülebilir. Engellerin siyah renk ile gösterildiği örnek ortam için algoritma başlangıç noktasından başlayarak sezgisel f(n) fonksiyonunun sonucuna göre ortamı tarayarak hedef noktasına ilerler. Tarama alanından hedef noktaya ulaştığında bir çıktı üretilir.

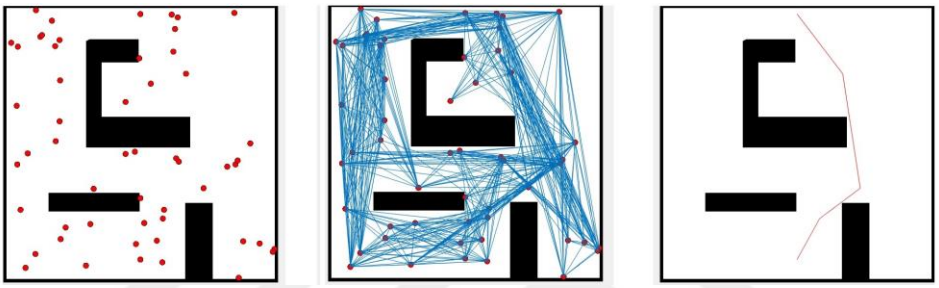
3.2. PRM Algoritması

1996 yılında yeni bir hareket planlama yöntemi olarak sunulan PRM algoritması iki aşamada ilerler: Öğrenme aşaması ve sorgu aşaması. Öğrenme aşamasında olasılıksal yol haritası, düğümleri çarpışma içermeyen konfigürasyonlara göre rastgele dağıtır. Daha sonra düğümler arasında yollar oluşturarak bunları grafik olarak saklar. Bu yollar basit ve hızlı yerel bir planlayıcı kullanılarak hesaplanır. Sorgu aşamasında robot herhangi bir başlangıç ve hedef noktası arasında yol haritasının iki düğümüne bağlanır; yol haritası daha sonra bu iki düğümü birleştiren bir yol için araştırılır. Yöntemin uygulanması kolaydır. Her türlü holonomik robota uygulanabilir. (Bir holonomik sistem kontrol edilebilir serbestlik derecelerinin toplam serbestlik derecelerine eşit olduğu zamandır.)

Let: $V \leftarrow \emptyset; E \leftarrow \emptyset;$
1: **loop**
2: $c \leftarrow a$ (uygun düğüm) C_{free}'de konfigüre edilmiş,
3: $V \leftarrow V \cup \{c\}$
4: $N_c \leftarrow a$ V'den seçilen bir dizi (uygun) düğüm,
5: **for all** $c' \in N_c$, c'ye olan mesafeyi artırmak için, **do**
6: **if** c' ve c G'ye bağlı değil **then**
7: **if** yerel planlayıcı c' ve c arasında bir yol bulur, then
8: c'c kenarını E'ye ekle

Şekil 3.3. PRM Algoritması

PRM algoritması Şekil 3.3'de özetlenmiştir (Geraerts & Overmars, 2004).



Şekil 3.4. PRM Algoritmasının örnek çıktısı

Şekil 3.4'de PRM algoritmasının örnek bir ortam için verdiği çıktı görülebilir. Engellerin siyah renk ile gösterildiği örnek ortam için belirlenen sayıda nokta algoritma tarafından ortama rastgele dağıtılır. Daha sonra bu noktalar arasında yollar oluşturulur ve başlangıç ile hedef nokta arasındaki en kısa yol çıktı olarak verilir.

3.3. Genetik Algoritma (GA)

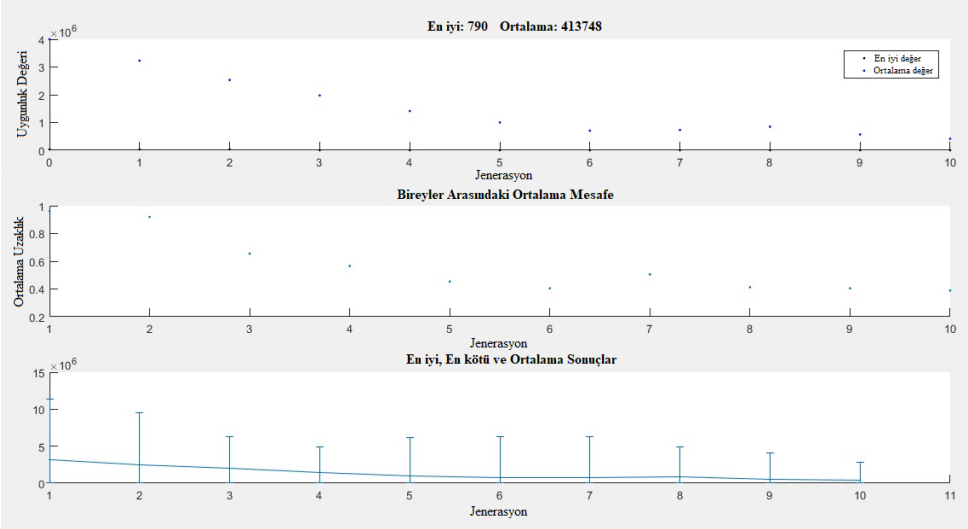
Genetik algoritmada kullanılan kavramlar, biyolojideki evrim teorisine benzer. Doğal yaşamın popülasyonları bireylerin bir arada yaşaması ile oluşur. GA algoritması için üretilen popülasyon, çok sayıda olası çözümün bir araya geldiği çok sayıda kişinin birleşmesiyle oluşur. Aday çözümler doğru şekilde kodlanmış matrislerde saklanır. Bu diziyi oluşturan her elemana birey denir ve her bir birey arama alanındaki belirli bir bölgeyi temsil eder.

Genetik algoritmada, başlangıçta ortaya çıkan bireyler genellikle rastgele üretilir, ancak bu zorunlu değildir. Özellikle çok sınırlı optimizasyon problemlerinde, başlangıçtaki bireyleri oluşturmak için tanımlanmış kısıtlamaların bazılarına dikkat edilerek daha iyi adaylar eğitilebilir. Popülasyon tarafından oluşturulan ilk genetik

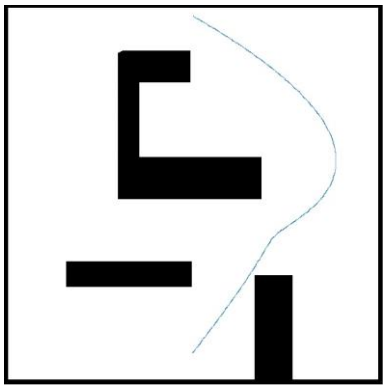
algoritma, seçim, çaprazlama ve mutasyon operatörleri olarak üç evrim operatörü ile birlikte çalışır. Genel olarak, bu operatörlerin her biri bir sonraki neslin her bir popülasyonuna uygulanır.

Seçim süreci popülasyonun alaka değerlerine dayanarak, yeni bireyler oluşturmak için ebeveyni seçme işlemidir. Geçiş operatörü, seçim işleminden sonra uygulanır ve ana bireye ait olan kromozomların belirli bölümlerinin karşılıklı yer değiştirmesini ve böylece yeni özelliklerde bireylerin oluşumunu ifade eder. Mutasyon işlemi, mutasyon olasılığına bağlı olarak, yeni oluşturulan bireyin kromozomlarından birinde bir geni değiştirme işlemidir.

Genetik algoritma işlemini sonlandırmanın birkaç yöntemi vardır: Algoritmanın çalışması sırasında istenen çözüm bulunduğunda; GA başlangıcında tanımlanan toplam yineleme sayısına ulaşıldığında veya yetenek değerinin sabit kalması durumunda. En iyi birey tarafından temsil edilen çözüm, soruna en uygun çözüm olarak sunulur ve algoritma durdurulur.



Şekil 3.5: Genetik Algoritma jenerasyonları



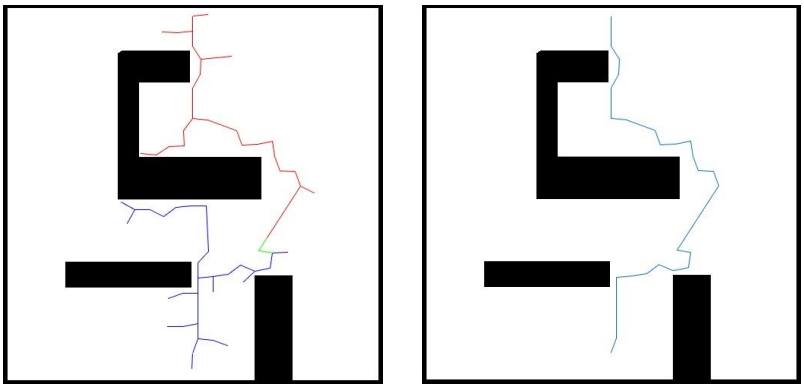
Şekil 3.6. Genetik Algoritmanın örnek çıktısı

Şekil 3.5'de ve Şekil 3.6'da Genetik Algoritmanın örnek ortam için verdiği çıktılar görülmektedir.

3.4. RRT Algoritması

1998 yılında Steven M. Lavalle tarafından yol planlama problemlerinin geniş bir sınıfı için rastgele data verileri tasarlanarak RRT (Hızlı Keşfeden Rastgele Ağaçlar) kavramı tanıtılmıştır. RRT'ler özellikle nonholonomik kısıtlamaları işlemek için tasarlanmıştır. RRT'ye kontrol girişi uygulanarak yinelemeli (iteratively) olarak genişletilir. PRM yaklaşımında olduğu gibi noktadan noktaya yakınsama gerektiren tekniklerin aksine sistem çok az rastgele seçilmiş noktalara doğru yönlendirilir. Birçok kez RRT ile holonomik, nonholonomik ve kinodinamik planlama sorunlarında 12. dereceye kadar serbestlik sağlanmıştır (Fizikte bir mekanik sistemin serbestlik derecesi (degree of freedom, DOF), konfigürasyonunu tanımlayan bağımsız parametrelerin sayısıdır. Fiziksel bir sistemin durumunu belirleyen parametrelerin sayısıdır ve makine mühendisliği, havacılık mühendisliği, robotik ve yapısal mühendislikteki vücut sistemlerinin analizinde önemlidir.).

RRT başlangıç düğümünden bitiş düğümüne yol alırken rastgele noktalar seçerek bu düğümlere dallanır ve hedef düğüme belli bir mesafede yaklaşıncaya dek bu işlemi tekrarlayan bir algoritmadır. RRT kendisinden önceki algoritmalardan farklı olarak bir sonraki düğüm seçiminde daha önce bulunmuş düğümleri işleme katmamaktadır. Yakındaki bir düğümü hesaba katmaktansa harita üzerinde rastgele bir düğüm seçilmekte ve bu düğüme yakın, daha evvel bulunmuş farklı bir düğümden hareket edilmektedir. Ağacın genişlemesi, başlangıçtan bitiş noktasına kadar tek yönlüdür. Öte yandan, ağacın iki yönlü bir şekilde genişlemesi fikri, tüm çözümün zaman verimliliğini arttırmaktadır. Bu teknik çift yönlü RRT (bRRT veya B-RRT) olarak adlandırılır. Ağaç hem başlangıç hem de hedef noktalara yerleştirilir ve sonra araştırılmamış kısımlara genişletilir. Bu iki ağacın birleşme noktası bulunduğunda ve algoritmanın çıktısı olan bu yol optimum bir yol olduğunda arama işlemi sonlandırılır.



Şekil 3.7. RRT Algoritmasının örnek çıktısı

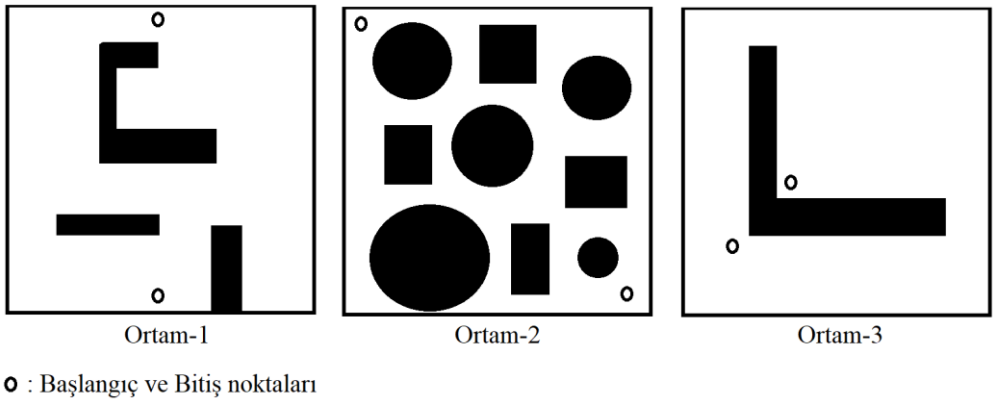
Şekil 3.7’de RRT algoritmasının örnek ortam için verdiği çıktı görülebilir.

4. ARAŞTIRMA SONUÇLARI VE TARTIŞMA

A* algoritmasının gerçek hayat uygulamalarına göre uzun sayılabilecek sürelerde çıktı vermesi, PRM algoritmasının hızlı sonuç üretmesine karşın kararlı sonuçlar üretememesi bu iki algoritma için belirlenen sorunlardır. Bu bölümde PRM algoritmasının kararlı çıktı verebilmesine ve A* algoritmasının daha hızlı sonuçlar üretebilmesine yönelik yapılan çalışmalar anlatılmıştır. Ayrıca A* algoritması ile gerçek bir şehir haritası üzerinde en kısa yol bulma uygulaması anlatılmıştır.

4.1. PRM Algoritmasını Kararlı Hale Getirmek için Ortamdaki Engellerin Köşe Noktalarına Düğüm Atamak

PRM algoritması çok hızlı sonuç üretse dahi aynı ortam, aynı başlangıç ve bitiş noktaları için bile olsa her çalıştırıldığında farklı sonuçlar üretebilmektedir. Bunun için yaptığımız çalışmada PRM algoritmasını çalıştırırken ortamda bulunan engellerin köşe noktaları koordinatları belirlenerek, o noktalara da düğüm ataması yaptırılmıştır. A* algoritmasının ürettiği çıktı, diğer algoritma sonuçları karşılaştırılırken optimum yol olarak ele alınmıştır. Çalışmada birbirinden farklı çeşitli tiplerde engellerin bulunduğu üç adet ortam yaratılmış, deneyler bu ortamlar üzerinde yapılmıştır.



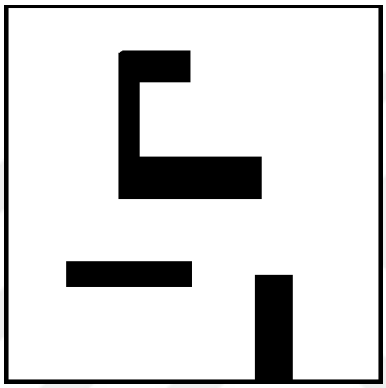
Şekil 4.1. Testler için belirlenen ortamlar.

Şekil 4.1'de farklı tipte engellere sahip ortamlar görülmektedir. Ortam-1 farklı tipte şekillere sahip dikdörtgen engellerden oluşmaktadır. Test aşamasında bu engel köşelerine düğüm atamaları yapılacaktır. Ortam-2'de köşesiz engellerin de bulunduğu bir ortam tercih edilmiştir. Böyle bir ortam için önerilen yöntemin nasıl kullanılabileceği gösterildi. Ortam-3'te ise başlangıç ve bitiş noktaları arasındaki

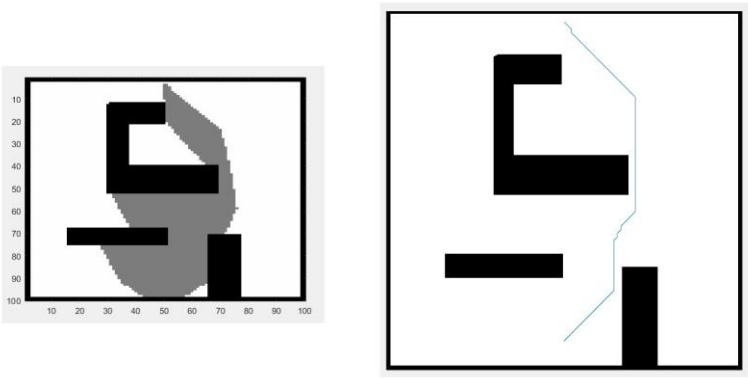
mesafe kuş uçuşu olarak çok az olduğu için tercih edilmiştir. Algoritmaların yol bulma aşamalarındaki davranışı incelenmiştir.

4.1.1. Ortam-1 İçin Elde Edilen Sonuçlar

Şekil 4.2'de Ortam-1 gösterilmiştir. Siyah alanlar ortamdaki engelleri temsil etmektedir. Tüm algoritma denemeleri için aynı başlangıç ve bitiş koordinatları kullanılmıştır. Ortam-1 için A Star algoritmasının verdiği çıktı Şekil 4.3'de gösterilmiştir.

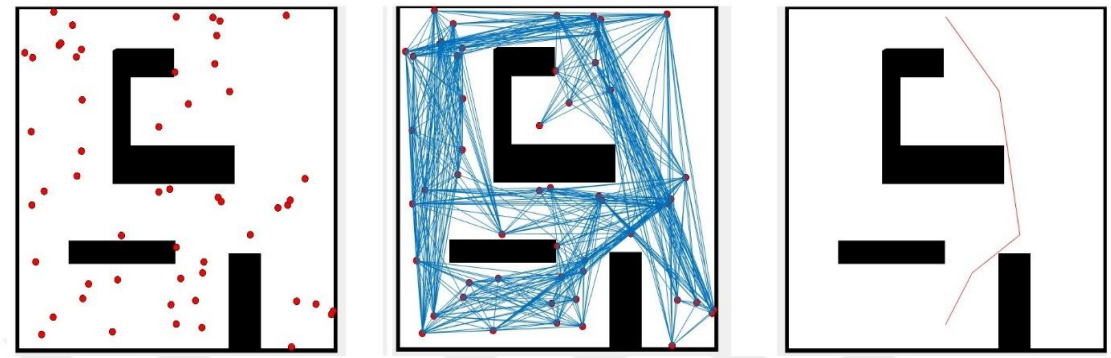


Şekil 4.2. Örnek olarak oluşturulan Ortam-1.

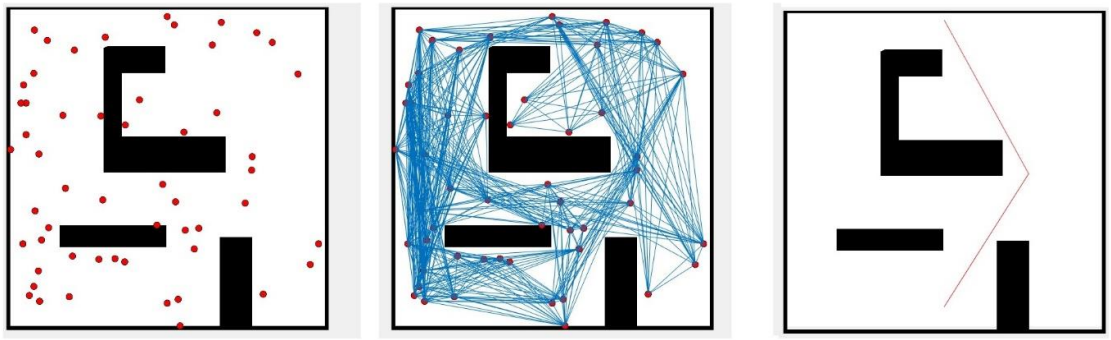


Şekil 4.3. Ortam-1 için A* algoritmasının verdiği çıktı.

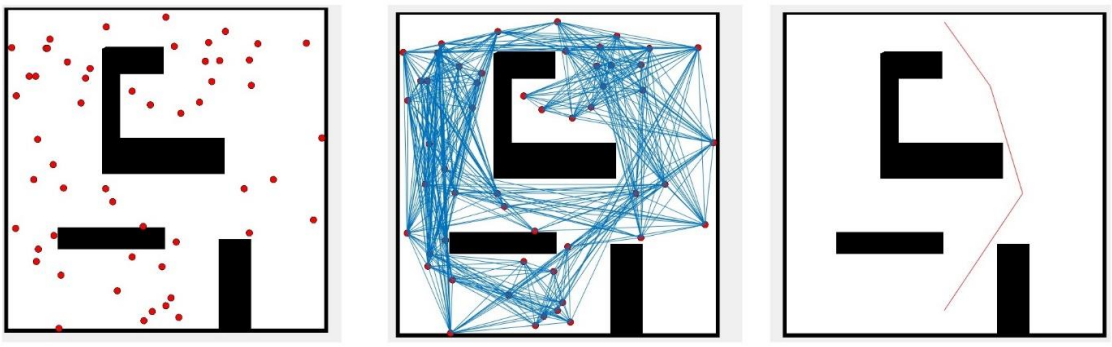
Ortam-1 için PRM algoritması öncelikle düğümleri rastgele dağıtacak şekilde çalıştırılmıştır. Çıktıların kararlığını test edebilmek için her durumda algoritma beş kez çalıştırılmıştır. Şekil 4.4, Şekil 4.5, Şekil 4.6, Şekil 4.7 ve Şekil 4.8'de bu denemelerin çıktısı verilmiştir.



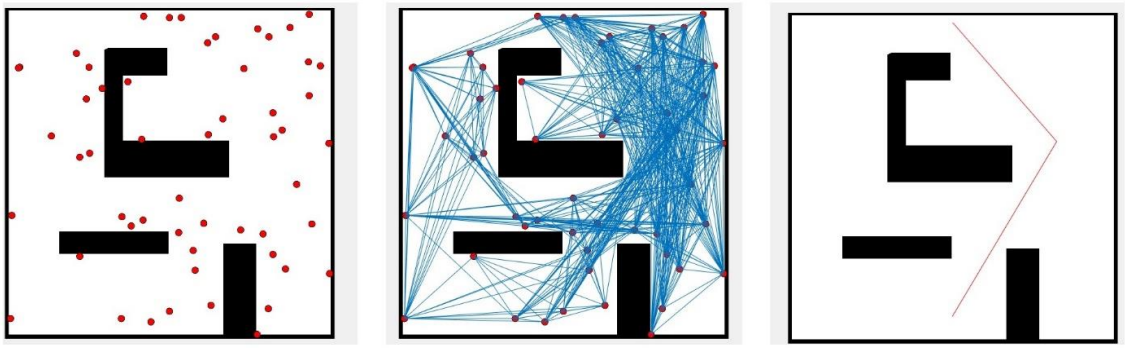
Şekil 4.4. Ortam-1 için PRM algoritmasının rastgele düğüm atayarak elde ettiği birinci sonuç.



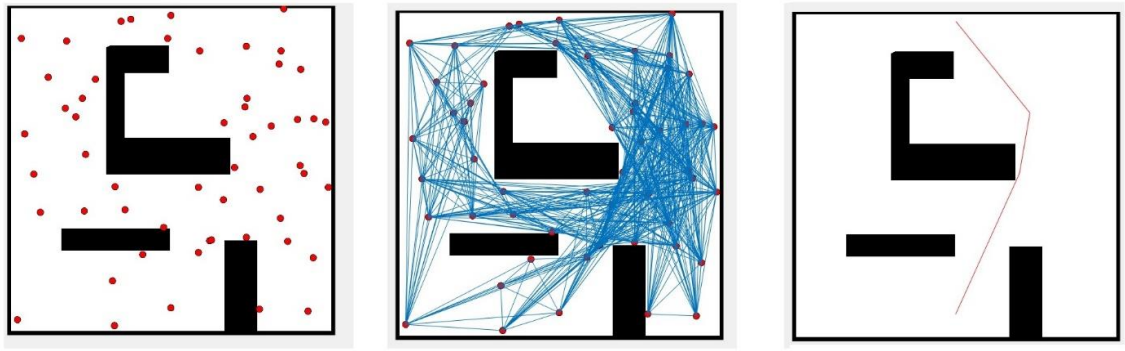
Şekil 4.5. Ortam-1 için PRM algoritmasının rastgele düğüm atayarak elde ettiği ikinci sonuç.



Şekil 4.6. Ortam-1 için PRM algoritmasının rastgele düğüm atayarak elde ettiği üçüncü sonuç.

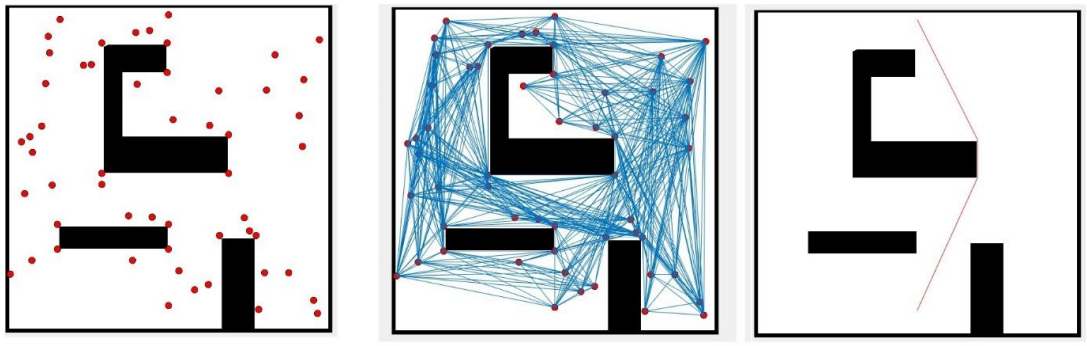


Şekil 4.7. Ortam-1 için PRM algoritmasının rastgele düğüm atayarak elde ettiği dördüncü sonuç.

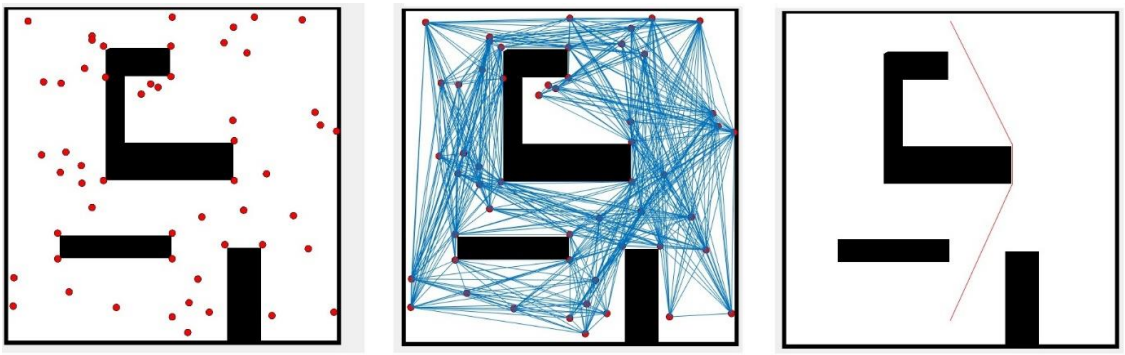


Şekil 4.8. Ortam-1 için PRM algoritmasının rastgele düğüm atayarak elde ettiği beşinci sonuç.

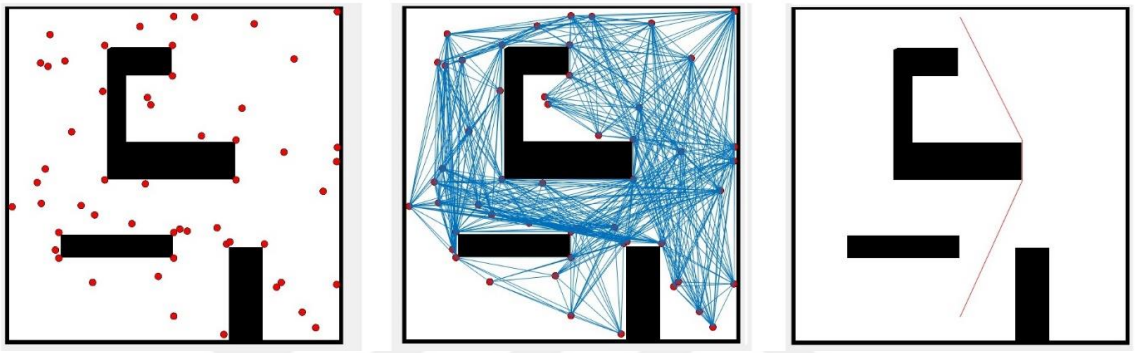
PRM algoritması düğümleri ortama rastgele olarak dağıtacak şekilde beş kez çalıştırıldıktan sonra görülebileceği gibi sonuçların hepsi birbirinden farklıdır. **Ortamda bulunan engellerin köşe noktalarına da düğüm eklenecek şekilde** beş kez daha çalıştırıldıktan sonra çıktılar Şekil 4.9, Şekil 4.10, Şekil 4.11, Şekil 4.12 ve Şekil 4.13'de verilmiştir.



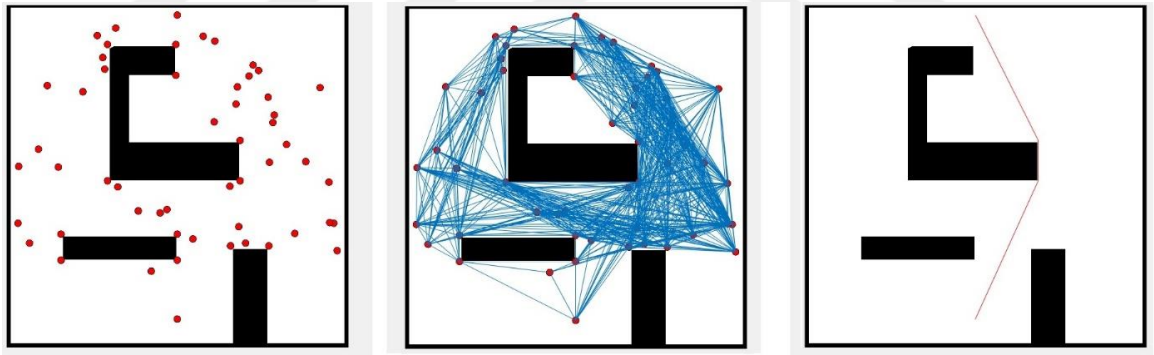
Şekil 4.9. Ortam-1 için PRM algoritmasının engel köşelerine düğüm atayarak elde ettiği birinci sonuç.



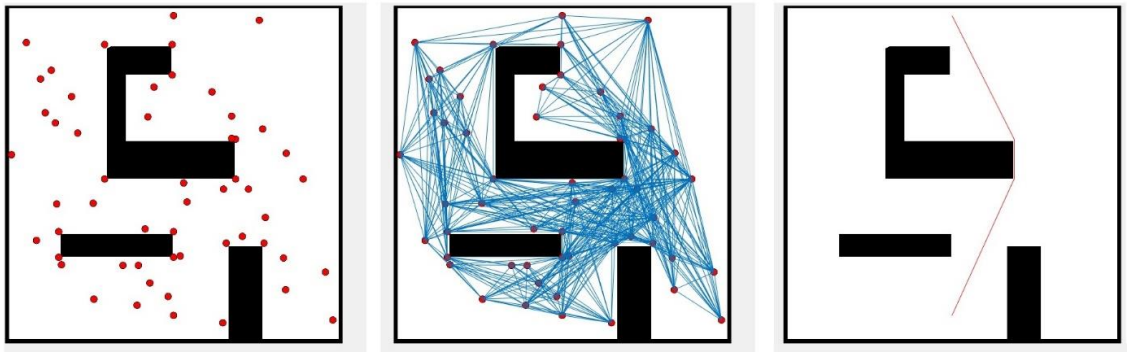
Şekil 4.10. Ortam-1 için PRM algoritmasının engel köşelerine düğüm atayarak elde ettiği ikinci sonuç.



Şekil 4.11. Ortam-1 için PRM algoritmasının engel köşelerine düğüm atayarak elde ettiği üçüncü sonuç.



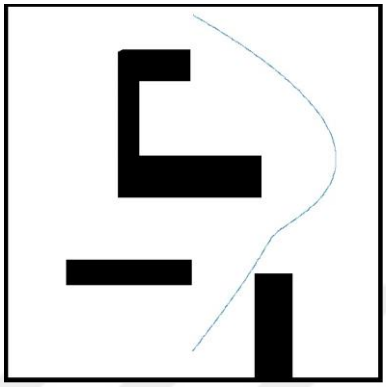
Şekil 4.12. Ortam-1 için PRM algoritmasının engel köşelerine düğüm atayarak elde ettiği dördüncü sonuç.



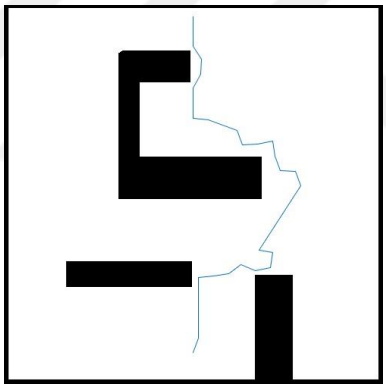
Şekil 4.13. Ortam-1 için PRM algoritmasının engel köşelerine düğüm atayarak elde ettiği beşinci sonuç.

Sonuçlardan görüleceği üzere ortamda bulunan engellerin köşe noktalarına da düğüm atandığı zaman beş sonucun hepsi birbiri ile aynı olmaktadır.

Aynı ortam için GA ve RRT algoritmasının verdiği çıktı Şekil 4.14 ve Şekil 4.15'de verilmiştir. A*, PRM, GA ve RRT algoritmalarından elde edilen sonuçlar tablo halinde Çizelge 4.1'de sunulmuştur.



Şekil 4.14. GA'nın birinci denemede ürettiği çıktı.



Şekil 4.15. RRT'nin birinci denemede ürettiği çıktı

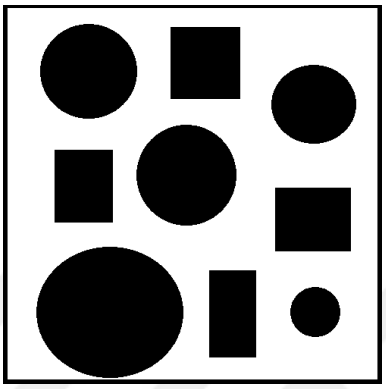
Çizelge 4.1. Ortam-1 için algoritma sonuçları

ORTAM-1	Deneme	Yol Uzunluğu	Süre (sn)	Ortalama Yol	Ortalama Süre (sn)
A*	1	527.8	57.7	527.8	57.7
	2	527.8	57.7		
	3	527.8	57.7		
	4	527.8	57.7		
	5	527.8	57.7		
PRM (Rastgele Düğüm)	1	506.7	3.0	515.7	2.4
	2	550.5	2.5		
	3	500.3	2.8		
	4	519.7	1.9		
	5	501.6	2.0		
PRM (Köşelere Düğüm)	1	485.7	2.4	485.7	2.4
	2	485.7	2.4		
	3	485.7	2.4		
	4	485.7	2.4		
	5	485.7	2.4		
RRT	1	684.0	3.5	676.6	5.92
	2	694.0	4.3		
	3	712.0	6.5		
	4	595.0	5.2		
	5	698.0	10.1		
GA	1	790.0	98.1	744.8	68.2
	2	735.0	54.3		
	3	717.0	61.3		
	4	803.0	69.0		
	5	679.0	58.3		

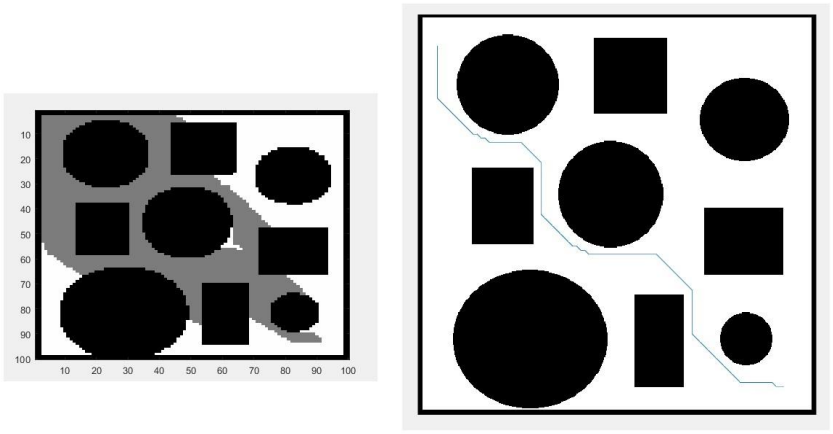
Çizelge 4.1'den görüleceği üzere A* algoritmasının Ortam-1 için ürettiği çıktının yol uzunluğu 527.8 birim iken süresi 57.7 saniye çıkmıştır. A* algoritması ile her çalıştırmada aynı sonuç elde edilebilmiştir. Ancak en iyi süre PRM algoritması ile elde edilmiştir. PRM algoritmasının engel köşelerine düğüm atıldığı haldeki çıktılarında kararlı hale gelmesi de yine tabloda görülmektedir. GA algoritması da A* algoritması gibi çıktı üretmek için uzun süreler harcamasına rağmen toplam yol uzunluğu yüksek kalmıştır. RRT algoritması da PRM algoritması gibi kısa sürede sonuç üretebilmiştir.

4.1.2. Ortam-2 İçin Elde Edilen Sonuçlar

Şekil 4.16'da Ortam-2 gösterilmiştir. Siyah alanlar ortamdaki engelleri temsil etmektedir. Tüm algoritma denemeleri için aynı başlangıç ve bitiş koordinatları kullanılmıştır. Ortam-2 için A Star algoritmasının verdiği çıktı Şekil 4.17'de gösterilmiştir.

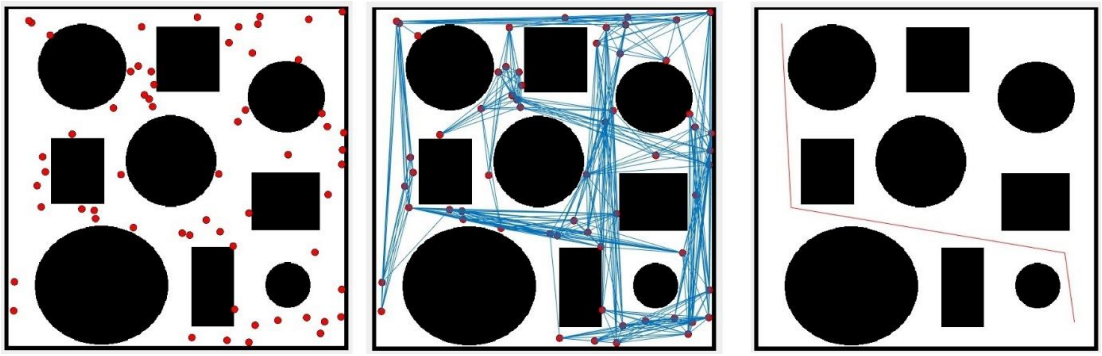


Şekil 4.16. Örnek olarak oluşturulan Ortam-2.

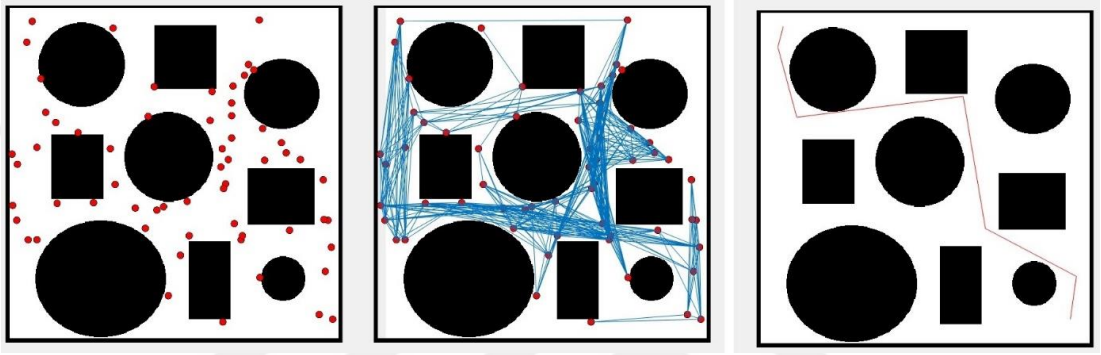


Şekil 4.17. Ortam-2 için A* algoritmasının verdiği çıktı.

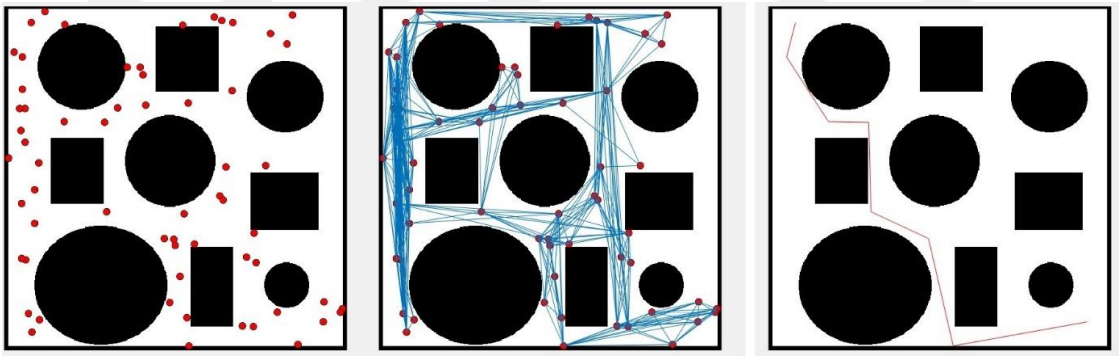
PRM algoritması öncelikle düğümleri ortama rastgele dağıtacak şekilde çalıştırılmıştır. Çıktıların kararlığını test edebilmek için her durumda algoritma beş kez çalıştırılmıştır. Şekil 4.18, Şekil 4.19, Şekil 4.20, Şekil 4.21 ve Şekil 4.22'de bu denemelerin çıktısı verilmiştir.



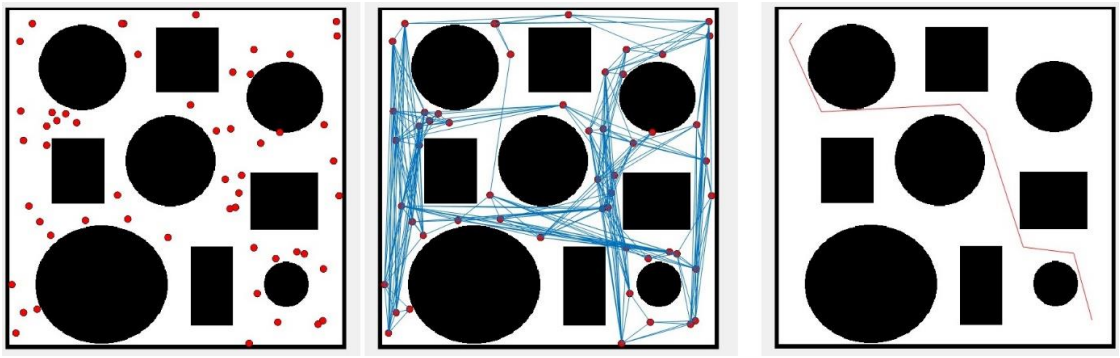
Şekil 4.18. Ortam-2 için PRM algoritmasının rastgele düğüm atayarak elde ettiği birinci sonuç.



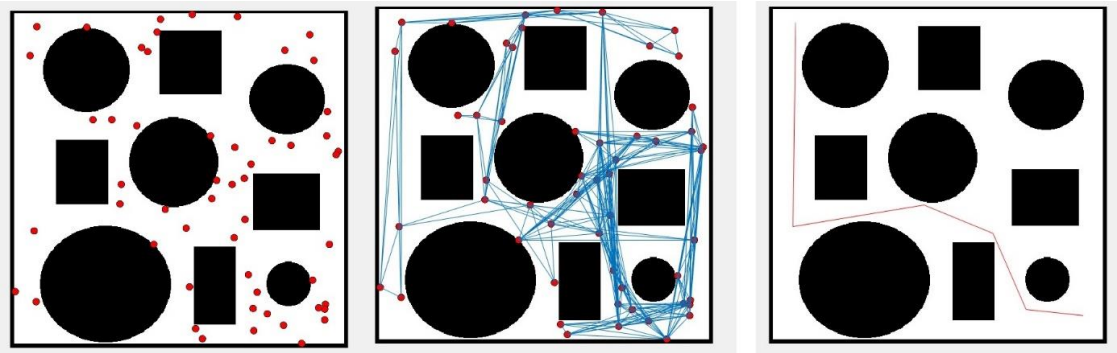
Şekil 4.19. Ortam-2 için PRM algoritmasının rastgele düğüm atayarak elde ettiği ikinci sonuç.



Şekil 4.20. Ortam-2 için PRM algoritmasının rastgele düğüm atayarak elde ettiği üçüncü sonuç.

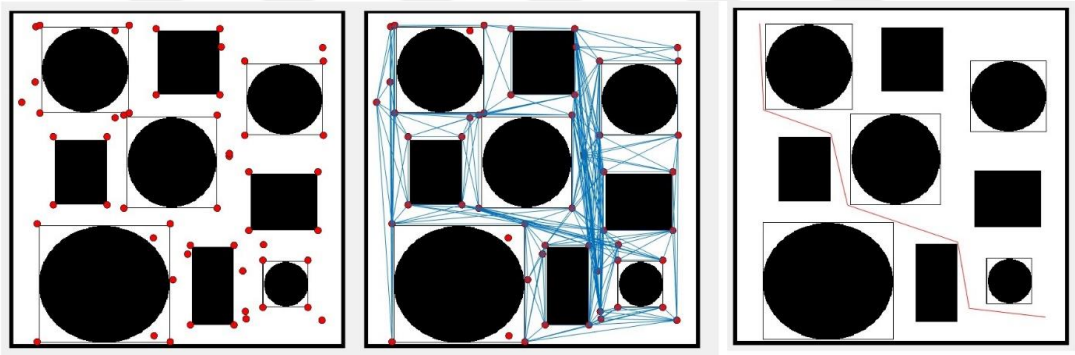


Şekil 4.21. Ortam-2 için PRM algoritmasının rastgele düğüm atayarak elde ettiği dördüncü sonuç.

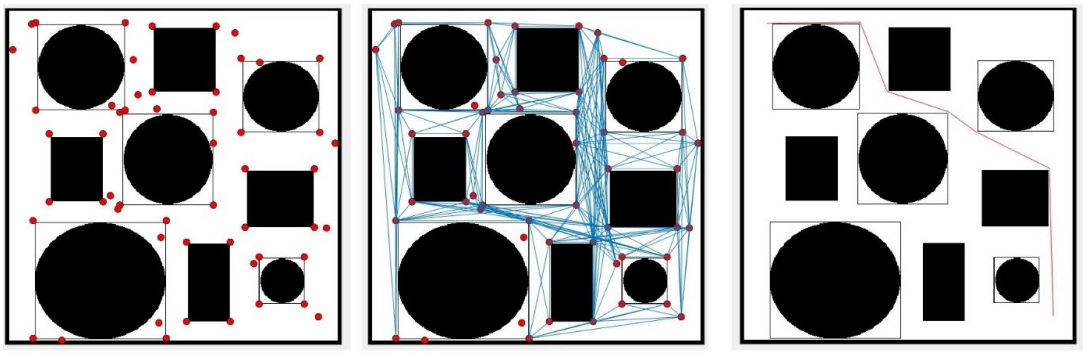


Şekil 4.22. Ortam-2 için PRM algoritmasının rastgele düğüm atayarak elde ettiği beşinci sonuç.

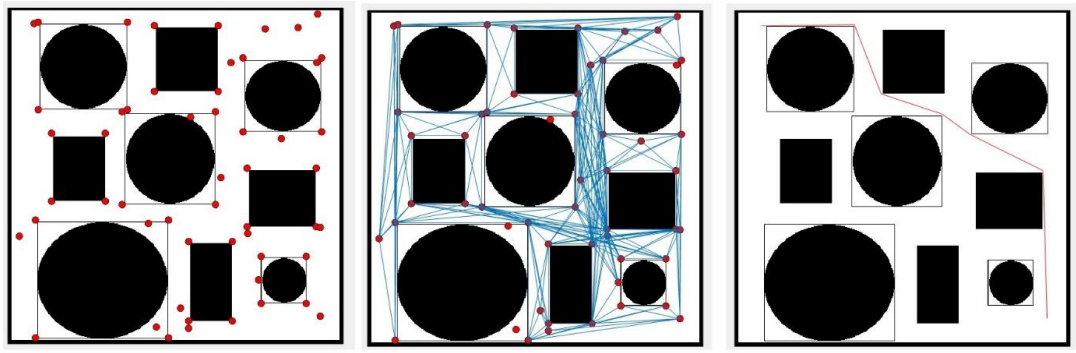
PRM algoritması düğümleri rastgele olarak dağıtacak şekilde beş kez çalıştırıldıktan sonra görülmektedir ki tüm sonuçlar birbirinden farklıdır. **Ortamda bulunan engellerin köşe noktalarına da düğüm eklenecek şekilde** beş kez daha çalıştırılmıştır. Ortamda köşesiz, yuvarlak vb. engel olması durumunda, bu engeller kare-dikdörtgen içine alınarak bu şekillerin köşe noktalarına düğüm ataması yapılmıştır. Şekil 4.23, Şekil 4.24, Şekil 4.25, Şekil 4.26 ve Şekil 4.27'de bu denemelerin çıktısı verilmiştir.



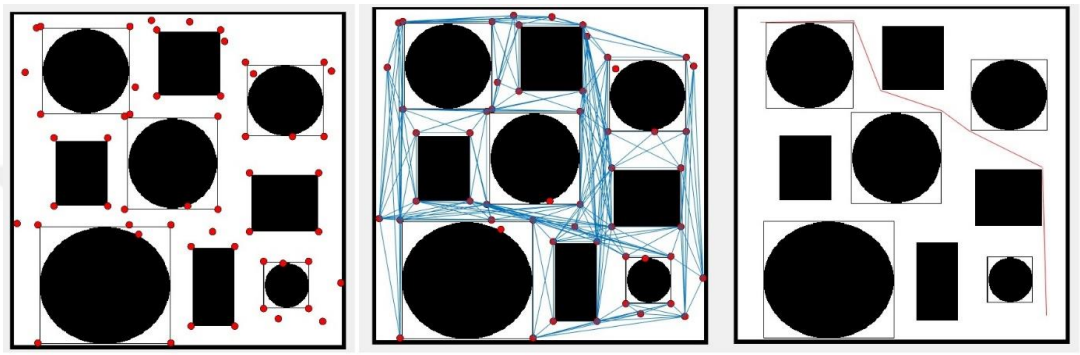
Şekil 4.23. Ortam-2 için PRM algoritmasının engel köşelerine düğüm atayarak elde ettiği birinci sonuç.



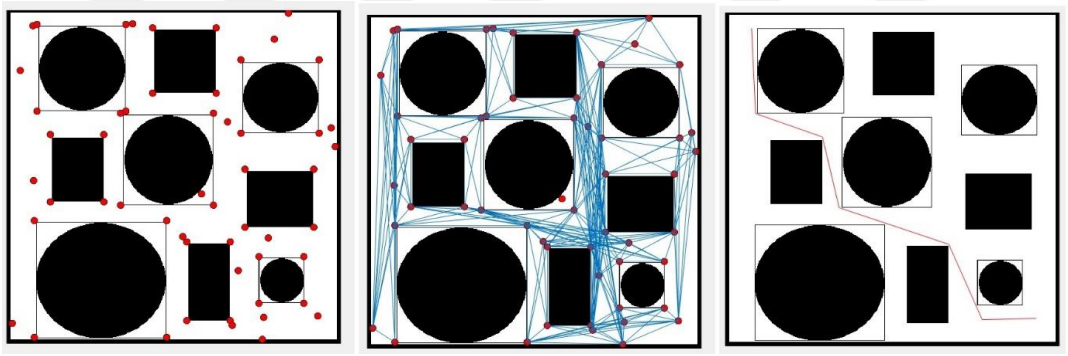
Şekil 4.24. Ortam-2 için PRM algoritmasının engel köşelerine düğüm atayarak elde ettiği ikinci sonuç.



Şekil 4.25. Ortam-2 için PRM algoritmasının engel köşelerine düğüm atayarak elde ettiği üçüncü sonuç.



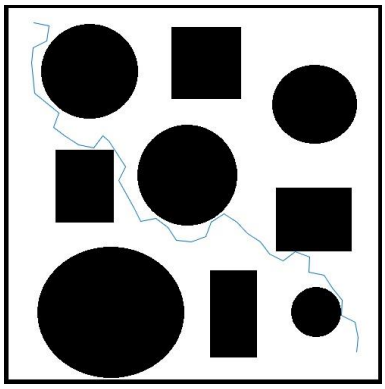
Şekil 4.26. Ortam-2 için PRM algoritmasının engel köşelerine düğüm atayarak elde ettiği dördüncü sonuç.



Şekil 4.27. Ortam-2 için PRM algoritmasının engel köşelerine düğüm atayarak elde ettiği beşinci sonuç.

Sonuçlardan görüleceği üzere ortamda bulunan engellerin köşe noktalarına da düğüm atandığı zaman beş sonucun üçü ve kalan ikisi birbiri ile aynı olmaktadır. İlk duruma göre daha iyi bir kararlılık yakalanmıştır.

Aynı ortam için RRT algoritmasının verdiği çıktı Şekil 4.28'da verilmiştir. GA bu ortam için çalıştırıldığında hata vererek sonuç üretememiştir. A*, PRM ve RRT algoritmalarından elde edilen sonuçlar tablo halinde Çizelge 4.2'de sunulmuştur.



Şekil 4.28. RRT'nin birinci denemede ürettiği çıktı.

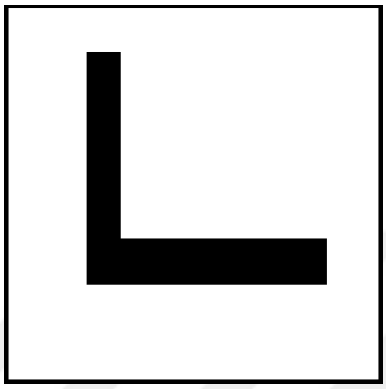
Çizelge 4.2. Ortam-2 için algoritma sonuçları

ORTAM-2	Deneme	Yol Uzunluğu	Süre (sn)	Ortalama Yol	Ortalama Süre (sn)
A*	1	719.4	68.6	719.4	68.6
	2	719.4	68.6		
	3	719.4	68.6		
	4	719.4	68.6		
	5	719.4	68.6		
PRM (Rastgele Düğüm)	1	833.9	2.6	797.4	1.96
	2	823.5	1.7		
	3	760.6	1.7		
	4	783.9	2.1		
	5	785.1	1.7		
PRM (Köşelere Düğüm)	1	713.3	1.8	728.0	2.1
	2	737.8	2.3		
	3	737.8	2.3		
	4	737.8	2.3		
	5	713.3	1.8		
RRT	1	840.0	2.7	828.0	2.24
	2	834.0	2.2		
	3	840.0	1.7		
	4	787.0	2.1		
	5	839.0	2.5		

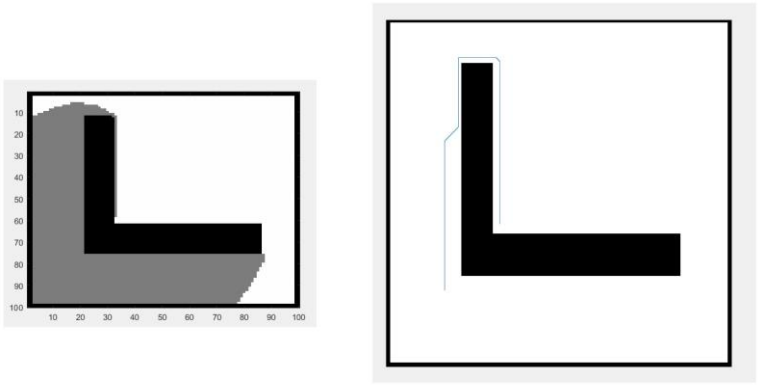
Çizelge 4.2'den görüleceği üzere A* algoritmasının Ortam-2 için ürettiği çıktının yol uzunluğu 719.4 birim iken süresi 68.6 saniye çıkmıştır. A* algoritması ile her çalışmada aynı sonuç elde edilmiştir. Ancak en iyi süre PRM ve RRT algoritmaları ile elde edilmiştir. PRM algoritmasının engel köşelerine düğüm atıldığı haldeki çıktılarında kararlı hale gelmesi de yine tabloda görülmektedir. RRT algoritması da PRM algoritması gibi kısa sürede sonuç üretebilmiştir.

4.1.3. Ortam-3 İçin Elde Edilen Sonuçlar

Şekil 4.29'da Ortam-3 gösterilmiştir. Siyah alanlar ortamdaki engelleri temsil etmektedir. Tüm algoritma denemeleri için aynı başlangıç ve bitiş koordinatları kullanılmıştır. Ortam-3 için A Star algoritmasının verdiği çıktı Şekil 4.30'da gösterilmiştir.

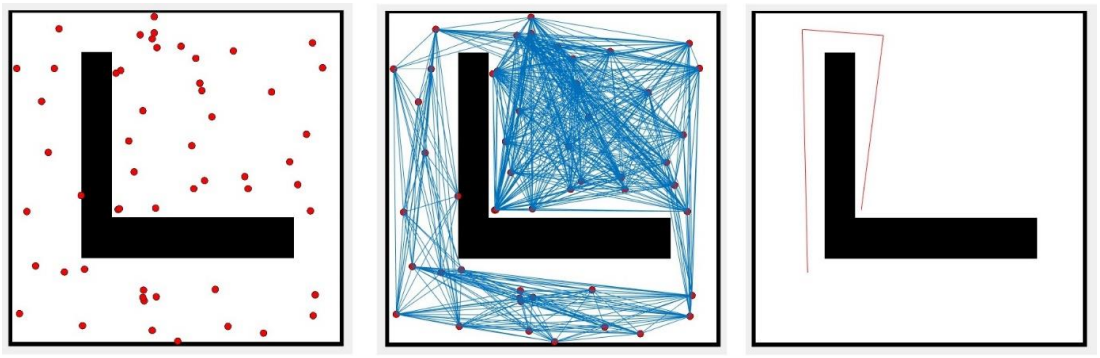


Şekil 4.29. Örnek için oluşturulan Ortam-3.

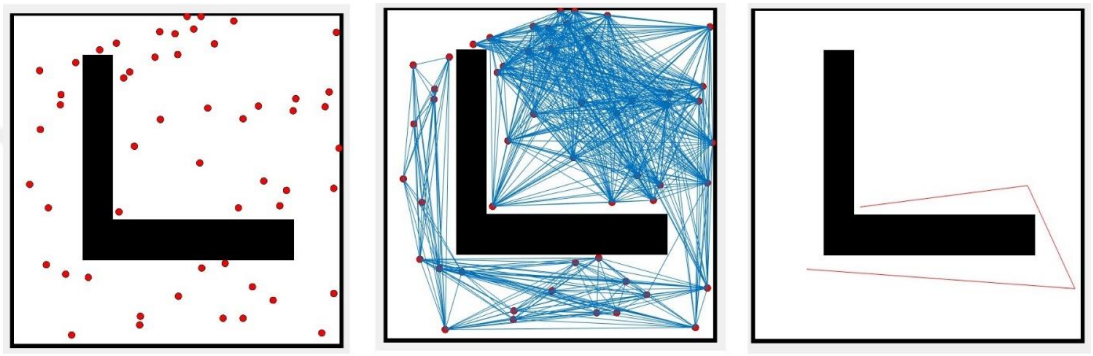


Şekil 4.30. Ortam-3 için A* algoritmasının verdiği çıktı.

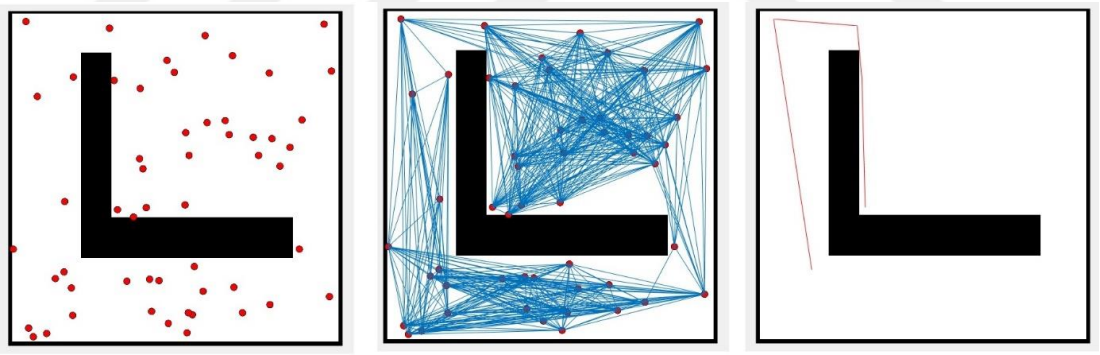
PRM algoritması öncelikle düğümleri ortama rastgele dağıtacak şekilde çalıştırılmıştır. Çıktıların kararlığını test edebilmek için her durumda algoritma beş kez çalıştırılmıştır. Şekil 4.31, Şekil 4.32, Şekil 4.33, Şekil 4.34 ve Şekil 4.35'de bu denemelerin çıktısı verilmiştir.



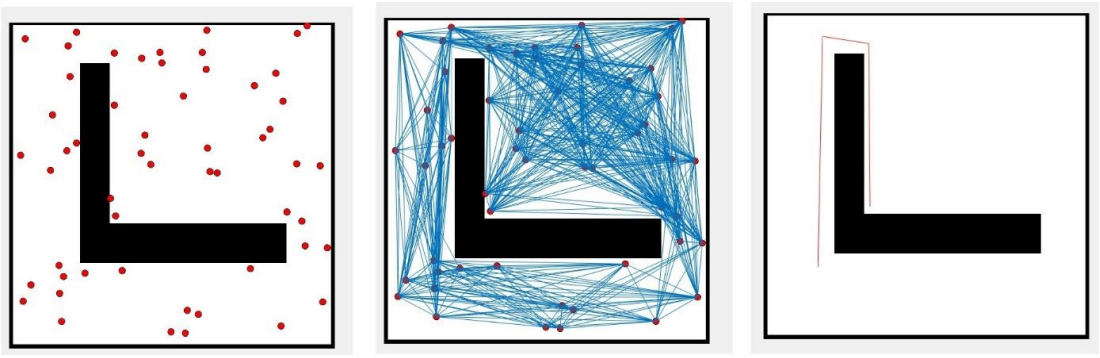
Şekil 4.31. Ortam-3 için PRM algoritmasının rastgele düğüm atayarak elde ettiği birinci sonuç.



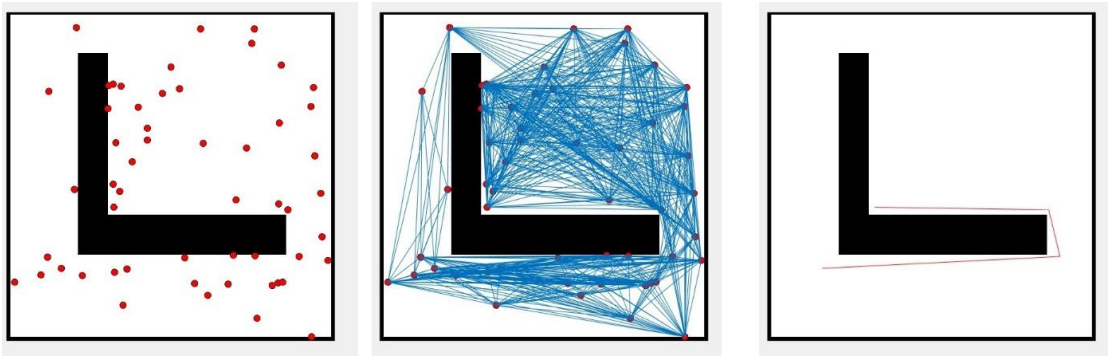
Şekil 4.32. Ortam-3 için PRM algoritmasının rastgele düğüm atayarak elde ettiği ikinci sonuç.



Şekil 4.33. Ortam-3 için PRM algoritmasının rastgele düğüm atayarak elde ettiği üçüncü sonuç.

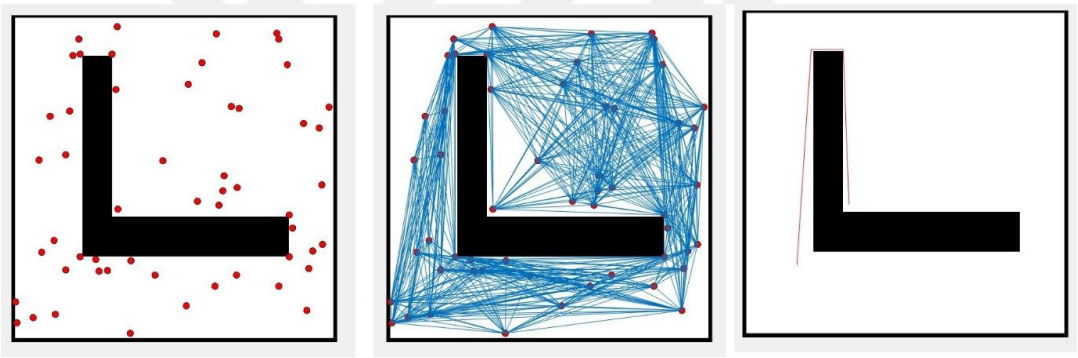


Şekil 4.34. Ortam-3 için PRM algoritmasının rastgele düğüm atayarak elde ettiği dördüncü sonuç.

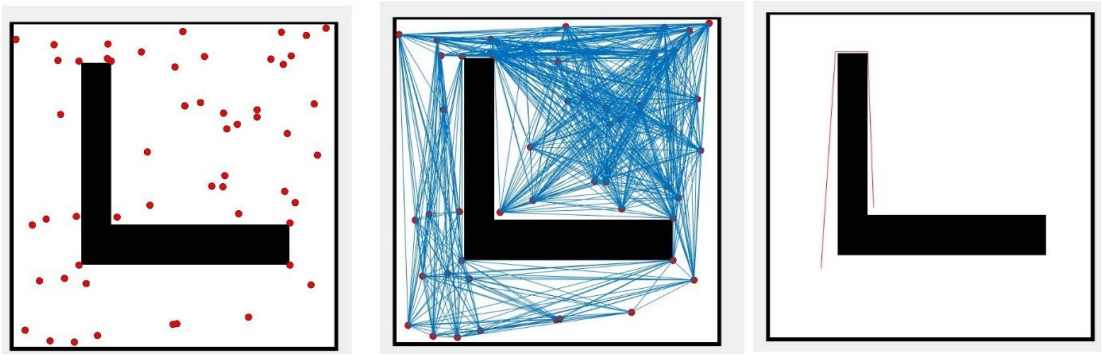


Şekil 4.35. Ortam-3 için PRM algoritmasının rastgele düğüm atayarak elde ettiği beşinci sonuç.

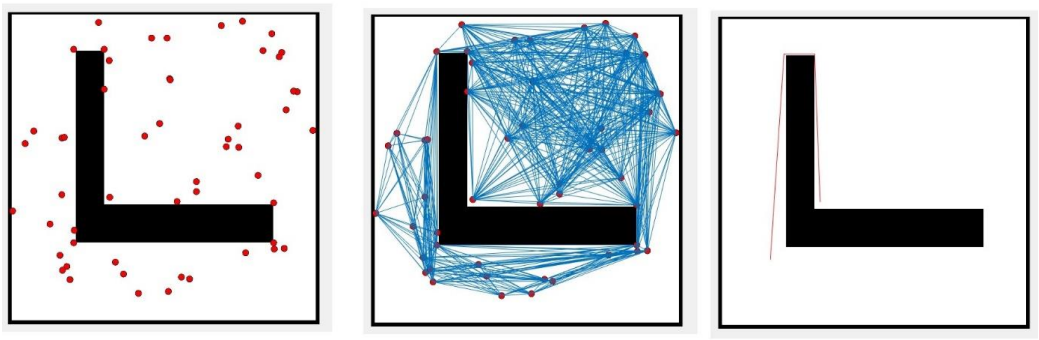
PRM algoritması düğümleri rastgele olarak dağıtacak şekilde beş kez çalıştırıldıktan sonra görülmektedir ki beş sonucun tamamı birbirinden farklıdır. **Ortamda bulunan engellerin köşe noktalarına da düğüm eklenecek şekilde** beş kez daha çalıştırıldıktan sonra çıktılar Şekil 4.36, Şekil 4.37, Şekil 4.38, Şekil 4.39 ve Şekil 4.40'da verilmiştir.



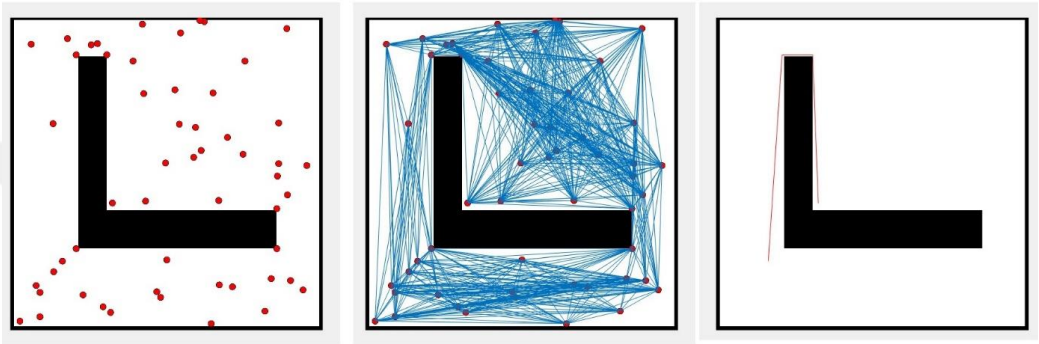
Şekil 4.36. Ortam-3 için PRM algoritmasının engel köşelerine düğüm atayarak elde ettiği birinci sonuç.



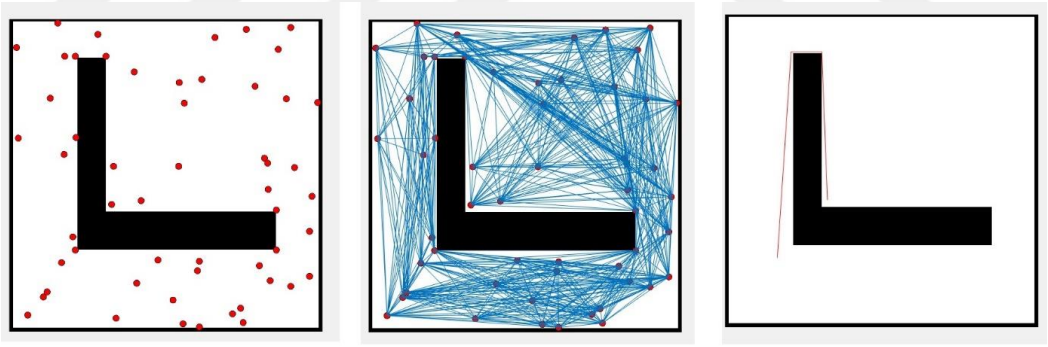
Şekil 4.37. Ortam-3 için PRM algoritmasının engel köşelerine düğüm atayarak elde ettiği ikinci sonuç.



Şekil 4.38. Ortam-3 için PRM algoritmasının engel köşelerine düğüm atayarak elde ettiği üçüncü sonuç.

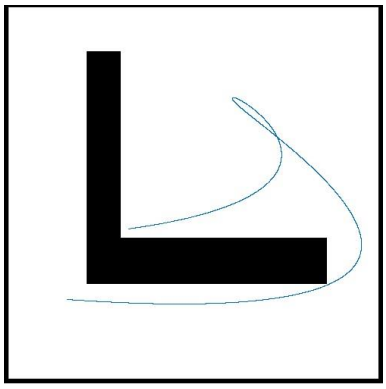


Şekil 4.39: Ortam-3 için PRM algoritmasının engel köşelerine düğüm atayarak elde ettiği dördüncü sonuç.

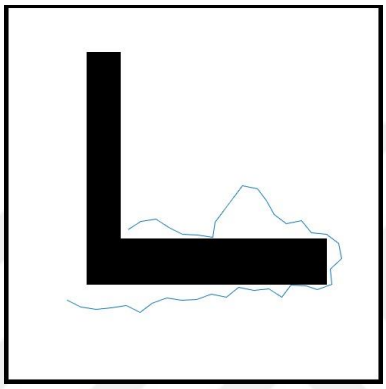


Şekil 4.40. Ortam-3 için PRM algoritmasının engel köşelerine düğüm atayarak elde ettiği beşinci sonuç.

Sonuçlardan görüleceği üzere ortamda bulunan engellerin köşe noktalarına da düğüm atandığı zaman beş sonucun hepsi birbiri ile aynı olmaktadır. Aynı ortam için GA ve RRT algoritmasının verdiği çıktı Şekil 4.41 ve Şekil 4.42'de verilmiştir. A*, PRM, GA ve RRT algoritmalarından elde edilen sonuçlar tablo halinde Çizelge 4.3'de sunulmuştur.



Şekil 4.41. GA'nın birinci denemede ürettiği çıktı.



Şekil 4.42. RRT'nin birinci denemede ürettiği çıktı.

Çizelge 4.3. Ortam-3 için algoritma sonuçları

ORTAM-3	Deneme	Yol Uzunluğu	Süre (sn)	Ortalama Yol	Ortalama Süre (sn)
A*	1	640.3	93.3	640.3	93.3
	2	640.3	93.3		
	3	640.3	93.3		
	4	640.3	93.3		
	5	640.3	93.3		
PRM (Rastgele Düğüm)	1	696.2	2.1	705.6	2.2
	2	730.8	3.0		
	3	712.6	2.0		
	4	688.1	2.0		
	5	700.7	2.3		
PRM (Köşelere Düğüm)	1	617.4	2.7	617.4	2.7
	2	617.4	2.7		
	3	617.4	2.7		
	4	617.4	2.7		
	5	617.4	2.7		
RRT	1	803.0	6.8	815.6	5.3
	2	853.0	5.5		
	3	834.0	3.8		
	4	790.0	5.8		
	5	798.0	4.8		
GA	1	1232.0	52.5	1117.2	60.0
	2	1143.0	73.2		
	3	973.0	51.7		
	4	1111.0	62.1		
	5	1127.0	60.9		

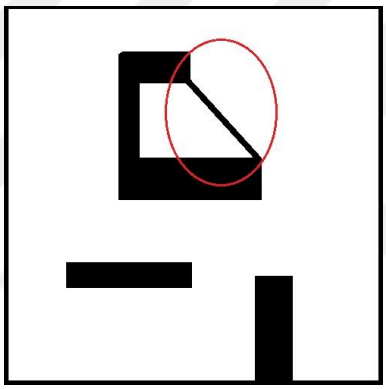
Çizelge 4.3'den görüleceği üzere A* algoritmasının Ortam-3 için ürettiği çıktının yol uzunluğu 640.3 birim iken süresi 93.3 saniye çıkmıştır. A* algoritması ile her çalışmada aynı sonuç elde edilmiştir. Ancak en iyi süre PRM algoritması ile elde edilmiştir. PRM algoritmasının engel köşelerine düğüm atıldığı haldeki çıktılarında kararlı hale gelmesi de yine tabloda görülmektedir. GA algoritması da A* algoritması gibi çıktı üretmek için uzun süreler harcamış buna rağmen yol uzunluğu yine de yüksek kalmıştır. RRT algoritması da PRM algoritması gibi kısa sürede sonuç üretebilmiştir.

4.2. A* Algoritmasının Daha Hızlı Çıktı Verebilmesi İçin Ortamdaki Açık Uçlu Engelleri Kapatmak

A* algoritmasının ürettiği sonuçların toplam maliyeti iyi olmakla beraber, çıktı için harcanan süre gerçek dünya için kabul edilebilir değildir. Yapılan çalışmada eğer başlangıç ve bitiş noktaları bu alanlarda değilse ortamdaki engellerin Şekil 4.43'de gösterildiği gibi açık uçları birleştirilerek, algoritmanın lüzum olmayan alanları taramasının önüne geçilmiş ve çalışma süresi düşürülmeye çalışılmıştır.

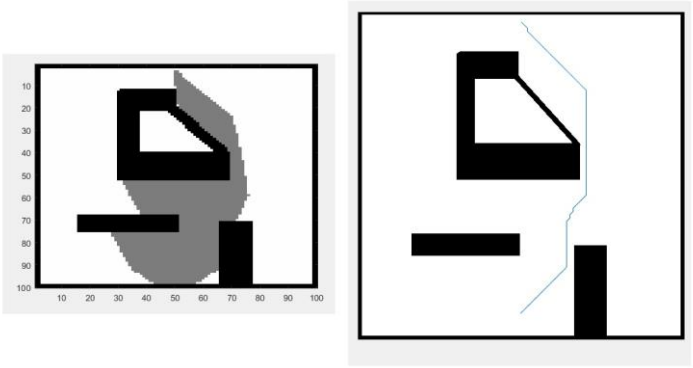
4.2.1. Ortam-1 için Elde Edilen Sonuçlar

Şekil 4.43'de Ortam-1'deki açık uçlu engelin kapatılmış hali verilmiştir.

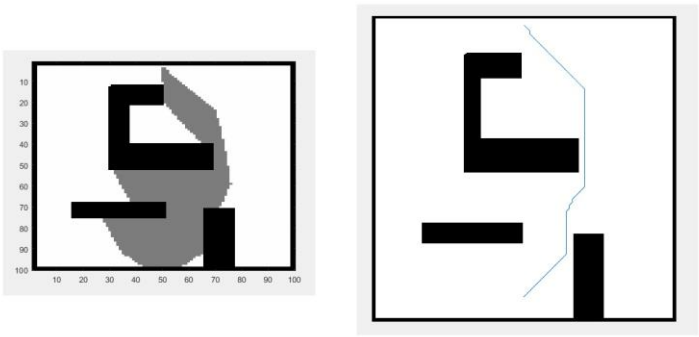


Şekil 4.43. Ortam-1'de açık uçlu engelin kapatılmış hali.

Şekil 4.44'de Ortam-1'deki açık uçlu engelin kapatılmış hali için A* algoritmasının verdiği çıktı gösterilmiştir.



Şekil 4.44. A* algoritmasının Ortam-1'de açık uçlu engellerin kapatıldığı haldeki çıktısı.



Şekil 4.45. A* algoritmasının Ortam-1 için çıktısı.

Şekil 4.45'de Ortam-1 için A* algoritmasının verdiği çıktı verilmiştir.

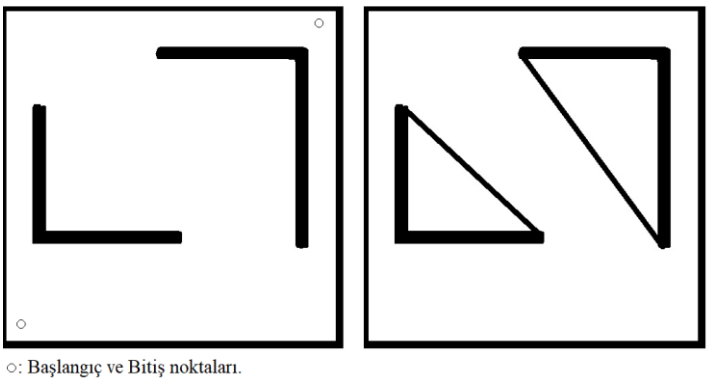
Çizelge 4.4. A* algoritmasının açık uçlu engellerin açık ve kapalı durumları için verdiği değerler.

ORTAM-1	Süre	Yol Uzunluğu
A* (Açık)	60.5	527.8
A* (Kapalı)	58.6	527.8

Açık uçlu engellerin açık ve kapatılmış halleri için A* algoritmasının verdiği sonuçlar Çizelge 4.4'de özetlenmiştir. Görüldüğü üzere açık uçlu engel kapatıldığı zaman algoritma aynı yolu daha hızlı şekilde bulmuştur.

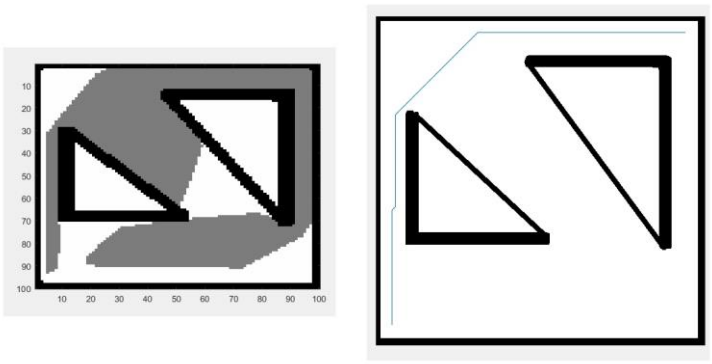
4.2.2. Ortam-4 için Elde Edilen Sonuçlar

Şekil 4.46'da Ortam-4'deki açık uçlu engelin açık ve kapatılmış hali verilmiştir.



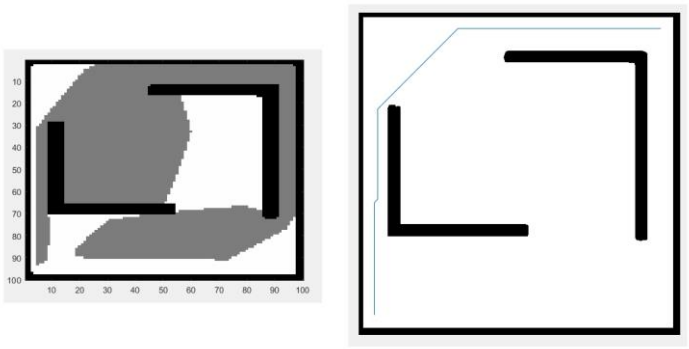
Şekil 4.46. Örnek olarak oluşturulan Ortam-4'teki açık uçlu engellerin açık ve kapalı halleri.

Şekil 4.47'de Ortam-4'deki açık uçlu engelin kapatılmış hali için A* algoritmasının verdiği çıktı gösterilmiştir.



Şekil 4.47. A* algoritmasının Ortam-4'de açık uçlu engellerin kapatıldığı haldeki çıktısı

Şekil 4.48'de Ortam-4 için A* algoritmasının verdiği çıktı verilmiştir.



Şekil 4.48. A* algoritmasının Ortam-4 için çıktısı.

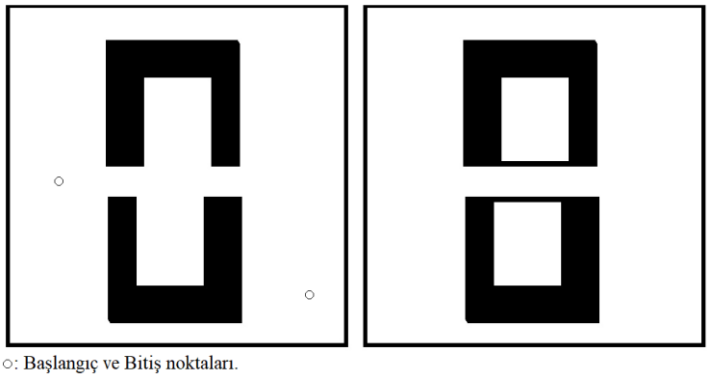
Çizelge 4.5. A* algoritmasının açık uçlu engellerin açık ve kapalı durumları için verdiği değerler.

ORTAM-4	Süre (sn)	Yol Uzunluğu
A* (Açık)	412	813.2
A* (Kapalı)	116	813.2

Açık uçlu engellerin açık ve kapatılmış halleri için A* algoritmasının verdiği sonuçlar Çizelge 4.5'de özetlenmiştir. Görüldüğü üzere açık uçlu engel kapatıldığı zaman algoritma aynı yolu daha hızlı şekilde bulmuştur. A* algoritmasının gereksiz olarak tarayarak süre harcadığı bölüm de Şekil 4.47 ve Şekil 4.48'de net olarak görülmektedir.

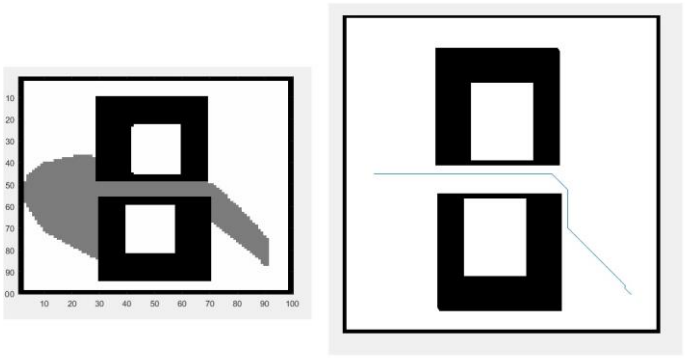
4.2.3. Ortam-5 için Elde Edilen Sonuçlar

Şekil 4.49'da Ortam-5'deki açık uçlu engellerin açık ve kapatılmış hali verilmiştir.



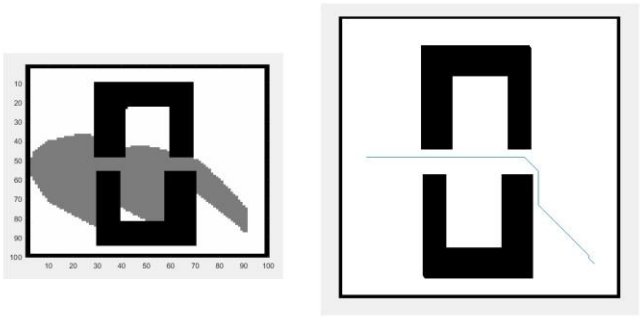
Şekil 4.49. Ortam-5 açık uçlu engellerin açık ve kapalı halleri.

Şekil 4.50'de Ortam-5'deki açık uçlu engelin kapatılmış hali için A* algoritmasının verdiği çıktı gösterilmiştir.



Şekil 4.50. A* algoritmasının Ortam-5'de açık uçlu engellerin kapatıldığı haldeki çıktısı

Şekil 4.51'de Ortam-5 için A* algoritmasının verdiği çıktı gösterilmiştir.



Şekil 4.51. A* algoritmasının Ortam-5 için çıktısı.

Çizelge 4.6. A* algoritmasının açık uçlu engellerin açık ve kapalı durumları için verdiği değerler.

ORTAM-5	Süre (sn)	Yol Uzunluğu
A* (Açık)	64.2	521.6
A* (Kapalı)	46.5	521.6

Açık uçlu engellerin açık ve kapatılmış halleri için A* algoritmasının verdiği sonuçlar Çizelge 4.6'da özetlenmiştir. Görüldüğü üzere açık uçlu engel kapatıldığı zaman algoritma aynı yolu daha hızlı şekilde bulmuştur. A* algoritmasının gereksiz olarak tarayarak süre harcadığı bölüm de Şekil 4.50 ve Şekil 4.51'de net olarak görülmektedir.

4.3. A* Algoritmasını Kullanarak Gerçek bir Ortam için En Kısa Yolu Bulmak

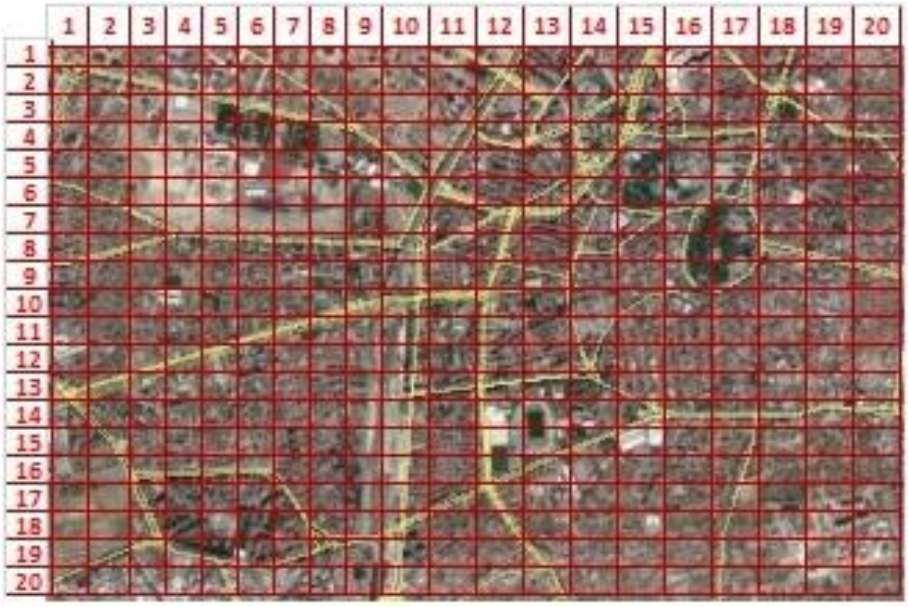
Ulaşım, geçmişten günümüze insanoğlunun hayatında büyük yer kaplayan bir kavram olmuştur. İlk insanlar yürüyerek, daha sonraki insanlar çeşitli hayvanları evcilleştirerek ulaşım ihtiyaçlarını karşılamışlardır. Tekerleğin icadıyla ilk taşıtların yapılması ve günümüzde de gelişen teknolojilerle birlikte otomobiller, uçaklar vb taşıtlar aracılığı ile ulaşım kavramı üst noktalara taşınmıştır. Artan nüfusa oranla büyüyen ulaşım ihtiyacını karşılamak için daha çok araç kullanılmaya başlandı, daha çok yol yapıldı ama bununla birlikte trafik yoğunluğu ve yol bulma gibi problemler de ortaya çıkmıştır. Bu sorunları en aza indirmek için çeşitli teknolojiler geliştirildi. Ulaşılacak hedefe giden en kısa ve en hızlı yolları bulmak için GPS destekli navigasyon sistemleri yapıldı. Bu çalışmada ise robotik sistemlerde sık kullanılan yol planlama algoritması A* kullanılarak bir aracın gideceği hedefe en kısa yolu bulması sağlanmıştır. Mevcut araçların kullanımına sunulmuş navigasyon sistemleri haricinde, gelecekte sıkça kullanılacağı öngörülen sürücüsüz araçlar, kendi kendisine ilerleyen robotlar vb. için mevcut harita paketlerinden parçalar alarak en kısa yolun bulunması

için robotik sistemlerde ve özellikle bilgisayar oyunları sektöründe sık kullanılan yol bulma algoritmaları kullanılacaktır. Bu çalışmada da basit olarak bir harita parçasının algoritma haritasına dönüştürülerek, başlangıç noktasından hedefe giden en kısa yolun bulunması sunulmuştur.

Google Haritalar üzerinden .jpeg dosya uzantılı olarak alınan harita parçası elle hücrelere ayrılarak algoritma haritası üzerine işlenmiştir. Örnek harita parçası Şekil 4.52'de verilmiştir.

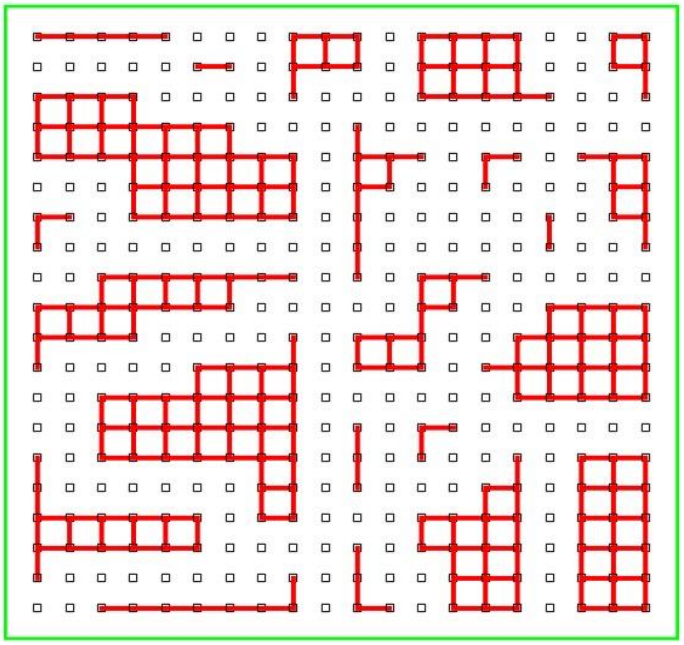
Şekil 4.52. Google Haritalar'dan örnek olarak alınan harita parçası.

Şekil 4.53'de harita parçasının hücrelere ayrılmış hali verilmiştir.



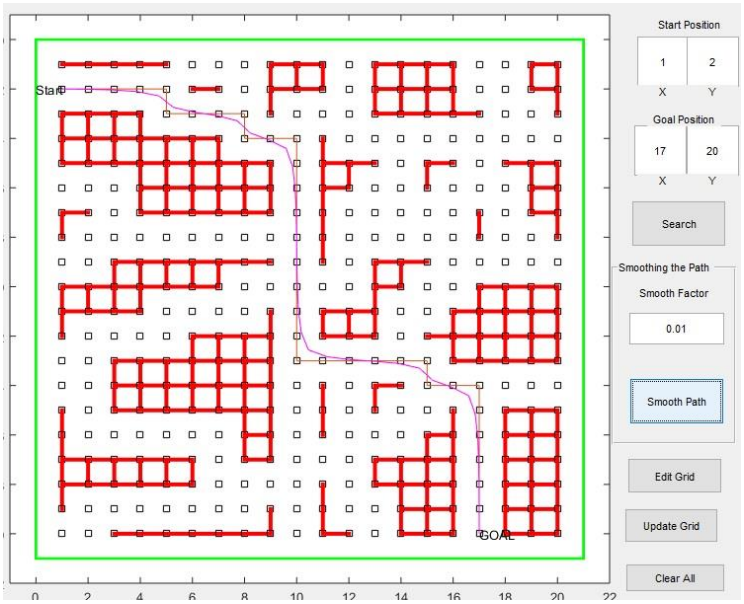
Şekil 4.53. Harita parçasının hücrelere ayrılması.

20x20 hücreden oluşacak şekilde gride ayrılan harita parçası üzerindeki aktif yolların tamamı, 20x20 büyüklüğündeki algoritma haritası üzerine işlenmiştir. Şekil 4.54'de hücrelere ayrılmış harita parçasının algoritma haritasına işlenmiş hali verilmiştir.



Şekil 4.54. Harita parçasının algoritmaya işlenmiş hali.

Google Haritalar parçası, Algoritma Haritası üzerinde işlendikten sonra Şekil 4.55'de gösterildiği şekilde A* algoritması başlangıç ve bitiş noktaları arasındaki en kısa yolu buldu.



Şekil 4.55. Başlangıç ve bitiş noktaları arasındaki en kısa yol.

Algoritma sonucu Şekil 4.56’da olduğu şekilde harita parçası üzerine işlenmiştir.

Şekil 4.56. Algoritma sonucu harita parçası üzerine işlenmiştir.

Şekil 4.56’da da sonucu görüldüğü üzere, A* algoritması kullanılarak gerçek bir ortam için en kısa yol bulunmuştur.

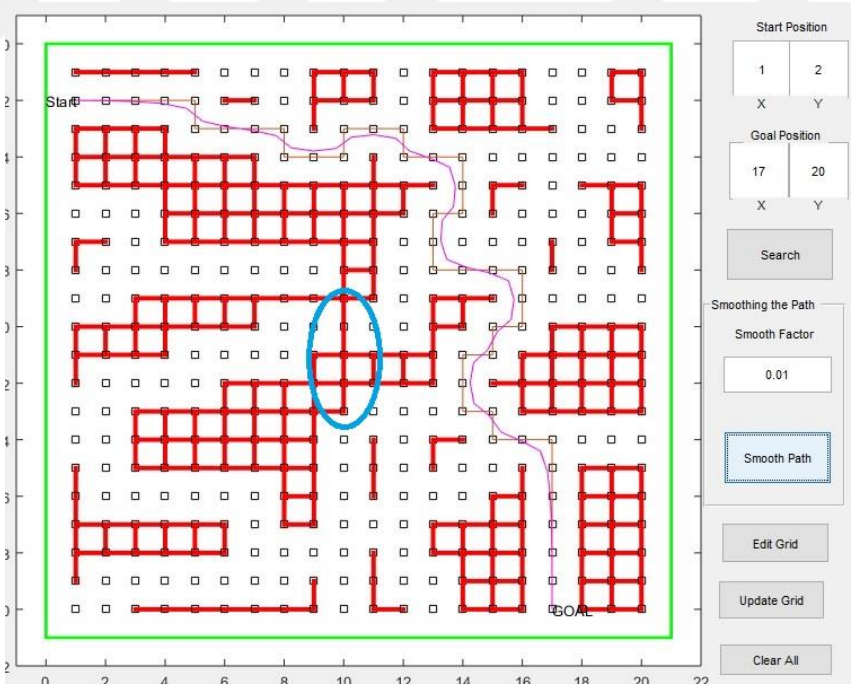
4.3.1. Ulaşımda Trafik Yoğunluğuna Göre En Kısa Yolun Bulunması

Bu adımın gerçekleştirilmesi için, Google Haritalar parçası üzerindeki aktif yolların trafik yoğunluğu bilgisinin verilmesi gerekmektedir. Bu bilgiye göre Algoritma Haritası üzerine işaretleme yapıldığı zaman, algoritma trafiğin yoğun olduğu bölgelerdeki yolları hesaplamaya katmayacak ve buna göre en kısa yolu bulacaktır. Bir nevi yoğun olan bölgeyi algoritmasında engel olarak işaretleyecek, alternatif yol bulacaktır.

Şekil 4.57’de harita parçası üzerinde trafiğin yoğun olduğu bölge gösterilmiştir.

Şekil 4.57. Trafiğin yoğun olduğu bölge.

Trafiğin yoğun olarak gösterildiği kısım, algoritma üzerinde engel olarak gösterilmiş, yeni güzergâh hesaplanarak Şekil 4.58'de gösterilmiştir.



Şekil 4.58. Trafik yoğunluğuna göre hesaplanan yeni rota.

Trafik yoğunluğuna göre A* algoritmasının hesapladığı yeni yol, Şekil 4.59'da gösterildiği şekilde harita parçası üzerine işlenmiştir.

Şekil 4.59. Yeni rotanın harita parçası üzerine işlenmiş hali.

Böylelikle A* algoritması ile gerçek bir ortam için trafik yoğunluğu bilgisine göre en kısa yol bulunmuş oldu.

5. SONUÇLAR VE ÖNERİLER

Bu tezde A* algoritması, PRM algoritması, GA ve RRT algoritması üzerinde çalışmalar yapılmıştır. Bu algoritmalar ile ilgili detaylı kaynak araştırması yapılmıştır. Algoritmalar tanıtılmış, oluşturulan örnek ortamlar için çıktıları elde edilerek sonuçları karşılaştırılmıştır.

A* algoritması bir ortamdaki başlangıç ve bitiş noktaları arasındaki en kısa yolu en iyi derecede bulabilmektedir. Ancak bu sonuç için algoritma uzun süreler harcamaktadır. Tez kapsamında yapılan çalışmada A* algoritmasının, ortamdaki gereksiz alanları taramasını engellemek için açık uçlu engeller kapatılmıştır. Paylaşılan tablolarda da görüleceği üzere A* algoritması aynı yol uzunluğunu daha düşük sürelerde bulabilmiştir.

PRM algoritması örnekleme tabanlı bir algoritma olarak çok hızlı sürelerde sonuç verebilmektedir. Ancak algoritmanın çalışma prensibine göre çalıştığı ortama rastgele düğüm ataması yapıldığı için ürettiği sonuçlar birbirinden farklı olmaktadır. Tez kapsamında çalışma yapılacak ortamda bulunan engellerin köşe noktaları referans kabul edilerek bu koordinatlara da düğüm ataması yapılmış, sonuçta PRM algoritmasının hep aynı sonuçları ürettiği görülmüştür.

A* ve PRM algoritmaları için çalışılan ortamlarda GA ve RRT algoritmalarının da ürettiği çıktılar verilmiştir. Böylece sezgisel sınıftaki A* algoritması, doğadan ilham alan algoritmalardan GA, klasik algoritmalardaki örnekleme tabanlı algoritmalar sınıfından PRM ve RRT algoritmaları karşılaştırılmış oldu. Örnekleme tabanlı algoritmaların çok hızlı sonuç verdiği görüldü. Temel GA'nın ise örnek ortamlar için hem uzun sürelerde çıktı verdiği hem de verdiği çıktıların toplam maliyetlerinin yüksek olduğu görüldü.

A* algoritması ile gerçek bir şehir haritasından alınan harita parçası üzerindeki iki noktanın en kısa yolu bulundu. Harita parçası gridlere ayrılarak, yol dışında kalan alanlar engel olarak işaretlendi. Bu şekilde algoritma haritası elde edildikten sonra algoritma çalıştırılarak en kısa yol bulundu. Ayrıca trafik yoğunluğuna göre de alternatif yol bulundu. Trafiğin yoğun olduğu yol, algoritma haritasında engel olarak işaretlendi ve algoritma mevcut durum için en kısa yolu buldu.

KAYNAKLAR

Amato, N. M., & Wu, Y. (1996). *A randomized roadmap method for path and manipulation planning.* Paper presented at the Proceedings of IEEE international conference on robotics and automation.

Aoude, G. S., Luders, B. D., Levine, D. S., & How, J. P. (2010). *Threat-aware path planning in uncertain urban environments.* Paper presented at the 2010 IEEE/RSJ International Conference on Intelligent Robots and Systems.

Barraquand, J., & Latombe, J.-C. (1991). Robot motion planning: A distributed representation approach. *The International Journal of Robotics Research, 10*(6), 628-649.

Benbouabdallah, K., & Qi-dan, Z. (2013). Genetic fuzzy logic control technique for a mobile robot tracking a moving target. *International Journal of Computer Science Issues (IJCSI), 10*(1), 607.

Bi, Z., Yimin, Y., & Wei, Y. (2008). *Hierarchical path planning approach for mobile robot navigation under the dynamic environment.* Paper presented at the 2008 6th IEEE International Conference on Industrial Informatics.

Branke, J. (2003). *Evolutionary approaches to dynamic optimization problems-introduction and recent trends.* Paper presented at the GECCO workshop on evolutionary algorithms for dynamic optimization problems.

Burns, B., & Brock, O. (2005). Sampling-based motion planning using predictive models.

Choset, H. M., Hutchinson, S., Lynch, K. M., Kantor, G., Burgard, W., Kavraki, L. E., & Thrun, S. (2005). *Principles of robot motion: theory, algorithms, and implementation*: MIT press.

Cui, S.-G., Wang, H., & Yang, L. (2012). *A simulation study of A-star algorithm for robot path planning.* Paper presented at the 16th international conference on mechatronics technology.

Desaraju, V. R., & How, J. P. (2011). *Decentralized path planning for multi-agent teams in complex environments using rapidly-exploring random trees.* Paper presented at the 2011 IEEE International Conference on Robotics and Automation.

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

Donald, B., Xavier, P., Canny, J., & Reif, J. (1993). Kinodynamic motion planning. *Journal of the ACM (JACM), 40*(5), 1048-1066.

Duchoň, F., Babinec, A., Kajan, M., Beňo, P., Florek, M., Fico, T., & Jurišica, L. (2014). Path planning with modified a star algorithm for a mobile robot. *Procedia Engineering, 96*, 59-69.

Elshamli, A., Abdullah, H. A., & Areibi, S. (2004). *Genetic algorithm for dynamic path planning.* Paper presented at the Canadian Conference on Electrical and Computer Engineering 2004 (IEEE Cat. No. 04CH37513).

Geraerts, R., & Overmars, M. H. (2004). A comparative study of probabilistic roadmap planners. In *Algorithmic Foundations of Robotics V* (pp. 43-57): Springer.

Goldberg, A. V., & Harrelson, C. (2005). *Computing the shortest path: A search meets graph theory.* Paper presented at the Proceedings of the sixteenth annual ACM-SIAM symposium on Discrete algorithms.

Goldberg, D. E. (1989). Genetic algorithms in search. *Optimization, and MachineLearning*.

Graf, B., Wandosell, J. H., & Schaeffer, C. (2001). *Flexible path planning for nonholonomic mobile robots.* Paper presented at the Proceedings of the Fourth European Workshop on Advanced Mobile Robots.

Gu, D.-W., Postlethwaite, I., & Kim, Y. (2006). *A comprehensive study on flight path selection algorithms.* Paper presented at the The IEE Seminar on Target Tracking: Algorithms and Applications 2006 (Ref. No. 2006/11359).

Harabor, D. (2012). Fast Pathfinding via Symmetry Breaking. In.

Harabor, D. D., & Grastien, A. (2011). *Online graph pruning for pathfinding on grid maps.* Paper presented at the Twenty-Fifth AAAI Conference on Artificial Intelligence.

Hart, P. E., Nilsson, N. J., & Raphael, B. (1968). A formal basis for the heuristic determination of minimum cost paths. *IEEE transactions on Systems Science and Cybernetics, 4*(2), 100-107.

Holland, J. (1975). Adaptation in natural and artificial systems: an introductory analysis. *Holland, JH*.

Hsu, D., Latombe, J.-C., & Kurniawati, H. (2006). On the probabilistic foundations of probabilistic roadmap planning. *The International Journal of Robotics Research, 25*(7), 627-643.

Hu, Y., & Yang, S. X. (2004). *A knowledge based genetic algorithm for path planning of a mobile robot.* Paper presented at the IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA'04. 2004.

Huang, H.-C., & Tsai, C.-C. (2011). *Global path planning for autonomous robot navigation using hybrid metaheuristic GA-PSO algorithm.* Paper presented at the SICE Annual Conference 2011.

Ismail, A., Sheta, A., & Al-Weshah, M. (2008). A mobile robot path planning using genetic algorithm in static environment. *Journal of Computer Science, 4*(4), 341-344.

Jung, I.-K., Hong, K.-B., Hong, S.-K., & Hong, S. C. (1999). *Path planning of mobile robot using neural network.* Paper presented at the ISIE'99. Proceedings of the IEEE International Symposium on Industrial Electronics (Cat. No. 99TH8465).

Karaman, S., & Frazzoli, E. (2011). Sampling-based algorithms for optimal motion planning. *The International Journal of Robotics Research, 30*(7), 846-894.

Karaman, S., Walter, M. R., Perez, A., Frazzoli, E., & Teller, S. (2011). *Anytime motion planning using the RRT.* Paper presented at the 2011 IEEE International Conference on Robotics and Automation.

Kavraki, L., Svestka, P., & Overmars, M. H. (1994). *Probabilistic roadmaps for path planning in high-dimensional configuration spaces* (Vol. 1994): Unknown Publisher.

Kennedy, J., & Eberhart, R. (1995). *Particle swarm optimization (PSO).* Paper presented at the Proc. IEEE International Conference on Neural Networks, Perth, Australia.

Kuffner Jr, J. J., & LaValle, S. M. (2000). *RRT-connect: An efficient approach to single-query path planning.* Paper presented at the ICRA.

Ladd, A. M., & Kavraki, L. E. (2004). Measure theoretic analysis of probabilistic path planning. *IEEE Transactions on Robotics and Automation, 20*(2), 229-242.

Lau, B., Sprunk, C., & Burgard, W. (2013). Efficient grid-based spatial representations for robot navigation in dynamic environments. *Robotics and Autonomous Systems, 61*(10), 1116-1130.

LaValle, S. M. (1998). Rapidly-exploring random trees: A new tool for path planning.
LaValle, S. M. (2006). *Planning algorithms*: Cambridge university press.

Lee, J., Kwon, O., Zhang, L., & Yoon, S.-E. (2014). A selective retraction-based RRT planner for various environments. *IEEE Transactions on Robotics, 30*(4), 1002-1011.

Liu, X., & Gong, D. (2011). *A comparative study of A-star algorithms for search and rescue in perfect maze.* Paper presented at the 2011 International Conference on Electric Information and Control Engineering.

Lozano-Pérez, T., & Wesley, M. A. (1979). An algorithm for planning collision-free paths among polyhedral obstacles. *Communications of the ACM, 22*(10), 560-570.

Lu, J., & Yang, D. (2007). *Path planning based on double-layer genetic algorithm.* Paper presented at the Third International Conference on Natural Computation (ICNC 2007).

Marble, J. D., & Bekris, K. E. (2013). Asymptotically near-optimal planning with probabilistic roadmap spanners. *IEEE Transactions on Robotics, 29*(2), 432-444.

Masehian, E., & Sedighizadeh, D. (2007). Classic and heuristic approaches in robot motion planning-a chronological review. *World Academy of Science, Engineering and Technology, 23*(5), 101-106.

McCourt, M., Ton, C. T., Mehta, S. S., & Curtis, J. W. (2016). *Adaptive step-length rrt algorithm for improved coverage.* Paper presented at the AIAA Guidance, Navigation, and Control Conference.

Nash, A., & Koenig, S. (2013). Any-angle path planning. *AI Magazine, 34*(4), 85-107.

Nazif, A. N., Davoodi, A., & Pasquier, P. (2010). Multi-agent area coverage using a single query roadmap: A swarm intelligence approach. In *Advances in practical multi-agent systems* (pp. 95-112): Springer.

Oleiwi, B. K., Al-Jarrah, R., Roth, H., & Kazem, B. I. (2014). *Multi objective optimization of trajectory planning of non-holonomic mobile robot in dynamic environment using enhanced ga by fuzzy motion control and a.* Paper presented at the International Conference on Neural Networks and Artificial Intelligence.

Paden, B., Nager, Y., & Frazzoli, E. (2017). *Landmark guided probabilistic roadmap queries.* Paper presented at the 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS).

Park, B., Choi, J., & Chung, W. K. (2012). *An efficient mobile robot path planning using hierarchical roadmap representation in indoor environment.* Paper presented at the 2012 IEEE International Conference on Robotics and Automation.

Pearce, R., Morales, M., & Amato, N. (2009). Structural improvement filtering strategy for prm. *Robotics: Science and Systems IV*, 167.

Ramirez-Serrano, A., & Boumedine, M. (1996). *Real-time navigation in unknown environments using fuzzy logic and ultrasonic sensing.* Paper presented at the Proceedings of the 1996 IEEE International Symposium on Intelligent Control.

Roberge, V., Tarbouchi, M., & Labonté, G. (2012). Comparison of parallel genetic algorithm and particle swarm optimization for real-time UAV path planning. *IEEE Transactions on industrial informatics, 9*(1), 132-141.

Saunders, J., Call, B., Curtis, A., Beard, R., & McLain, T. (2005). Static and dynamic obstacle avoidance in miniature air vehicles. In *Infotech@ Aerospace* (pp. 6950).

Slowik, A., & Bialko, M. (2007). *Design and optimization of IIR digital filters with non-standard characteristics using particle swarm optimization algorithm.* Paper presented at the 2007 14th IEEE International Conference on Electronics, Circuits and Systems.

Thomaz, C. E., Pacheco, M. A. C., & Vellasco, M. M. B. (1999). *Mobile robot path planning using genetic algorithms.* Paper presented at the International Work-Conference on Artificial Neural Networks.

Tu, J., & Yang, S. X. (2003). *Genetic algorithm based path planning for a mobile robot.* Paper presented at the 2003 IEEE International Conference on Robotics and Automation (Cat. No. 03CH37422).

Urmson, C., & Simmons, R. (2003). *Approaches for heuristically biasing RRT growth.* Paper presented at the Proceedings 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2003)(Cat. No. 03CH37453).

Wang, H., Zhou, J., Zheng, G., & Liang, Y. (2014). *HAS: Hierarchical A-Star algorithm for big map navigation in special areas.* Paper presented at the 2014 5th International Conference on Digital Home.

Yan, Z., Jouandeau, N., & Cherif, A. A. (2013). ACS-PRM: Adaptive cross sampling based probabilistic roadmap for multi-robot motion planning. In *Intelligent Autonomous Systems 12* (pp. 843-851): Springer.

Yang, K., & Sukkarieh, S. (2008). *3D smooth path planning for a UAV in cluttered natural environments.* Paper presented at the 2008 IEEE/RSJ International Conference on Intelligent Robots and Systems.

Yang, S. X., Hu, Y., & Meng, M. Q.-h. (2006). *A knowledge based GA for path planning of multiple mobile robots in dynamic environments.* Paper presented at the 2006 IEEE Conference on Robotics, Automation and Mechatronics.

Yap, P. K. Y., Burch, N., Holte, R. C., & Schaeffer, J. (2011). *Any-angle path planning for computer games.* Paper presented at the Seventh Artificial Intelligence and Interactive Digital Entertainment Conference.

Yun, S. C., Parasuraman, S., & Ganapathy, V. (2011). *Dynamic path planning algorithm in mobile robot navigation.* Paper presented at the 2011 IEEE Symposium on Industrial Electronics and Applications.

Zhang, H., Butzke, J., & Likhachev, M. (2012). *Combining global and local planning with guarantees on completeness.* Paper presented at the 2012 IEEE International Conference on Robotics and Automation.

Zu, W., Fan, G., Gao, Y., Ma, Y., Zhang, H., & Zeng, H. (2018). *Multi-UAVs Cooperative Path Planning Method based on Improved RRT Algorithm.* Paper presented at the 2018 IEEE International Conference on Mechatronics and Automation (ICMA).

Zucker, M., Kuffner, J., & Bagnell, J. A. (2008). Adaptive workspace biasing for sampling based planners.

ÖZGEÇMİŞ

KİŞİSEL BİLGİLER

Adı Soyadı : Muhammed Esat DERE

Uyruğu : T.C.

Doğum Yeri ve Tarihi : Meram, 1988

Telefon : 530 061 8016

E-mail : muhesad@gmail.com

EĞİTİM

Derece		Adı, İlçe, İl	Bitirme Yılı
Lise	:	Başak Lisesi	2005
Üniversite	:	Fırat Üniversitesi	2010

İŞ DENEYİMLERİ

Yıl	Kurum	Görevi
2012-2014	Seiso Enerji A.Ş.	Ar-Ge Mühendisi
2014-2017	Edites Mühendislik Ltd. Şti.	Kurucu Ortak
2015-	Efta Enerji Ltd. Şti.	Kurucu Ortak

UZMANLIK ALANI

Yazılım, Enerji.

YABANCI DİLLER

İngilizce

YAYINLAR

DERE E., DURDU A., (2004) Usage of the A* Algorithm to Find the Shortest Path in Transportation Systemts. In: International Conference on Advanced Technologies, Computer Engineering and Science (ICATCES), May 11-13, 2018 Safranbolu, Turkey