Merve Sevim

230202035

Robotlar için Matematik Temelleri

PROJE 3/4 - 4/4 Filo Yönetim Yazılımı

1. Giriş

Mobil robotların aynı ortam içerisinde eş zamanlı olarak görev yapması, günümüz otonom sistemlerinde sıklıkla karşılaşılan bir problemdir. Depo otomasyonu, arama-kurtarma sistemleri ve akıllı ulaşım gibi birçok uygulamada birden fazla robotun aynı harita üzerinde güvenli ve verimli şekilde hareket etmesi beklenmektedir. Ancak robot sayısı arttıkça, yol planlama süreci yalnızca hedefe ulaşmayı değil; aynı zamanda robot-robot etkileşimlerini, çakışmaları ve olası deadlock durumlarını da dikkate almak zorunda kalmaktadır.

Çok robotlu sistemlerde en temel zorluklardan biri, robotların aynı hücreye yönelmesi veya dar alanlarda karşılıklı olarak birbirlerini bloke etmesidir. Bu tür durumlar yalnızca basit çarpışmalara değil, aynı zamanda sistemin tamamen kilitlenmesine yol açabilen deadlock senaryolarına da neden olabilmektedir. Bu nedenle, güvenilir bir çok robotlu planlama sistemi; çakışmaları algılayabilmeli, deadlock durumlarını tespit edebilmeli ve uygun çözüm stratejileri üretebilmelidir.

Bu projede, çoklu robotların aynı ızgara tabanlı harita (binary occupancy grid) üzerinde hareket ettiği bir Fleet Manager sistemi MATLAB App Designer kullanılarak geliştirilmiştir. Sistem, farklı harita üretim modları, seçilebilir yol planlama algoritmaları, çakışma ve deadlock yönetimi mekanizmaları ile zaman adımlı bir simülasyon altyapısı sunmaktadır. Kullanıcı, arayüz üzerinden harita türünü, robot sayısını ve planlayıcıyı seçebilmekte; simülasyon sürecini gerçek zamanlı olarak izleyebilmektedir.

Geliştirilen uygulamada, rastgele ancak kontrol edilebilir şekilde üretilen karmaşık haritalar ve bilinçli olarak tasarlanmış dar koridor senaryoları kullanılarak farklı zorluk seviyelerinde test ortamları oluşturulmuştur. Böylece hem çakışmasız senaryolar hem de karşılıklı geçiş ve deadlock durumlarının ortaya çıktığı senaryolar sistematik olarak incelenebilmiştir.

Ayrıca her robot için doğrusal ve açısal hızlar ile diferansiyel sürüş modeline dayalı tekerlek hızları simülasyon sırasında hesaplanmış ve grafikler halinde sunulmuştur. Bu sayede yalnızca yol bulma başarımı değil, robotların kinematik davranışları da analiz edilebilmiştir.

Bu raporda, geliştirilen sistemin arayüz tasarımı, harita üretim yöntemleri, çok robotlu planlama yaklaşımı, çakışma ve deadlock yönetimi ile simülasyon sonuçları detaylı olarak açıklanmaktadır.

2. Sistem Mimarisi ve Arayüz Tasarımı

2.1 Genel Sistem Mimarisi

Geliştirilen Fleet Manager uygulaması, çok robotlu planlama ve simülasyon süreçlerini bütüncül bir yapı altında toplayan modüler bir mimariye sahiptir. Sistem; harita üretimi, robot ve hedef yerleşimi, yol planlama, çakışma ve deadlock yönetimi ile simülasyon ve analiz modüllerinden oluşmaktadır. Tüm bu bileşenler, MATLAB App Designer kullanılarak geliştirilen grafiksel kullanıcı arayüzü (GUI) üzerinden kontrol edilmektedir.

Uygulama, ızgara tabanlı bir ortamda çalışmakta olup harita bilgisi `binaryOccupancyMap` yapısı ile temsil edilmektedir. Bu yapı, engel ve serbest alanların hücre bazlı olarak tanımlanmasına olanak tanımakta ve yol planlama algoritmaları için ortak bir altyapı sunmaktadır. Harita oluşturulduktan sonra robotlar bu ortam içerisine yerleştirilmekte ve her robot için başlangıç ve hedef noktaları belirlenmektedir.

Sistem mimarisi, planlama ve simülasyon süreçlerini birbirinden ayıracak şekilde tasarlanmıştır. İlk aşamada seçilen planlayıcıya göre her robot için global yol planlaması gerçekleştirilmekte, ardından zaman adımlı simülasyon sürecinde robotlar bu yolları takip ederek hareket etmektedir. Simülasyon sırasında robotların anlık konumları, hız bilgileri ve etkileşim durumları sürekli olarak güncellenmektedir.

Çok robotlu yapılarda kritik öneme sahip olan çakışma ve deadlock durumları, simülasyon süreci içerisinde ayrı bir izleme mekanizması ile ele alınmaktadır. Robotlar arası bekleme ilişkileri `wait-for` grafiği mantığı ile modellenmekte ve görsel olarak kullanıcıya sunulmaktadır. Tespit edilen deadlock durumları için önceliklendirme, geri çekilme ve yeniden planlama gibi çözüm stratejileri uygulanmaktadır.

2.2 Kullanıcı Arayüzü Tasarımı

Geliştirilen kullanıcı arayüzü, Fleet Manager uygulamasının tüm temel işlevlerini tek bir ekran üzerinden erişilebilir olacak şekilde tasarlanmıştır. Arayüz, MATLAB App Designer'ın grid tabanlı yerleşim yapısı kullanılarak düzenlenmiş ve kullanıcı etkileşimini kolaylaştıracak biçimde bölümlere ayrılmıştır.

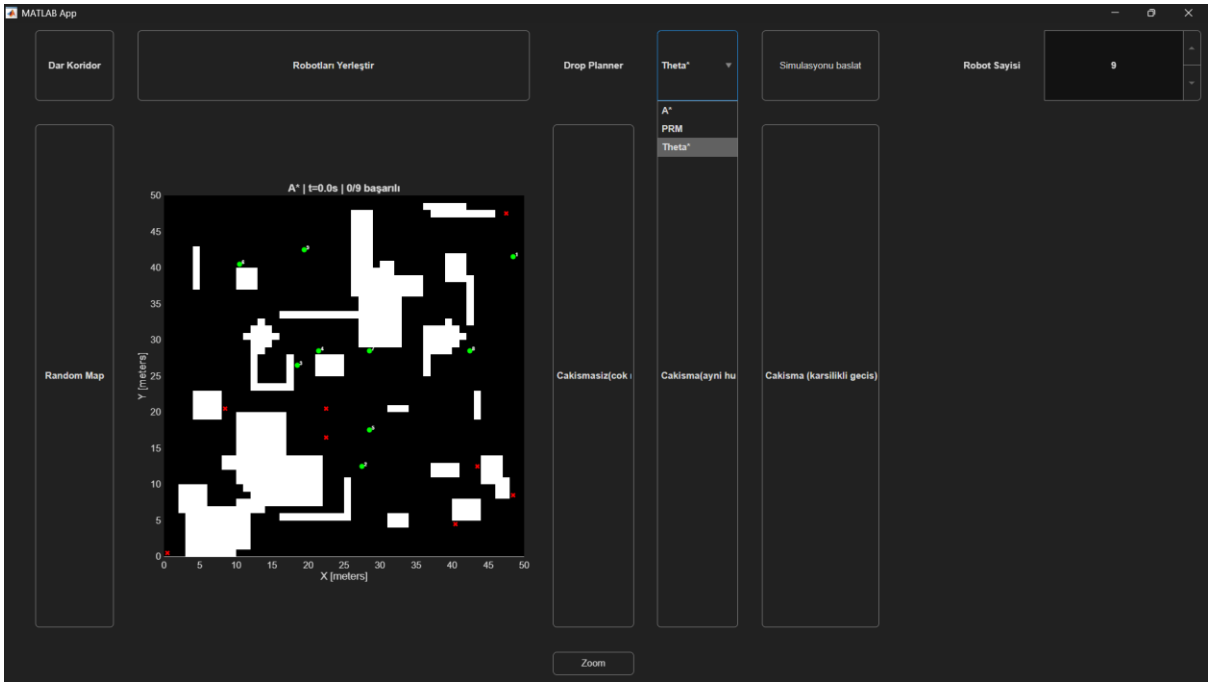
Arayüzün sol bölümünde harita üretimine yönelik kontroller yer almaktadır. Kullanıcı, rastgele harita (Random Map) veya dar koridor (Dar Koridor) modlarını seçerek farklı zorluk seviyelerine sahip senaryolar oluşturabilmektedir. Bu yapı, çakışmasız ve deadlock içeren senaryoların bilinçli olarak test edilmesine olanak tanımaktadır.

Üst bölümde, robot yerleştirme, planlayıcı seçimi ve simülasyon kontrolüne yönelik bileşenler bulunmaktadır. Kullanıcı, robot sayısını spinner aracılığıyla belirleyebilmekte ve A*, PRM veya Theta* gibi farklı yol planlama algoritmaları arasında seçim yapabilmektedir. Planlayıcı seçimi değiştirildiğinde, yol planlama süreci yeniden başlatılarak aynı senaryonun farklı algoritmalarla karşılaştırılması mümkün hale gelmektedir.

Merkezde yer alan görselleştirme alanı (UIAxes), harita, robotlar, hedefler, planlanan rotalar ve gerçekleşen yolların eş zamanlı olarak izlenmesini sağlamaktadır. Planlanan yollar kesikli çizgilerle, robotların simülasyon sırasında izlediği gerçek yollar ise düz çizgilerle gösterilmektedir. Ayrıca robotların birbirleriyle olan bekleme ilişkileri, oklarla gösterilen wait-for grafiği aracılığıyla görselleştirilmektedir.

Arayüz üzerinde simülasyon süresi, tamamlanan robot sayısı ve deadlock durumları gibi kritik metrikler dinamik olarak güncellenmektedir. Deadlock tespit edildiğinde kullanıcı, arayüz üzerinden görsel ve metinsel uyarılar ile bilgilendirilmektedir.

Şekil 2.1



Şekil 2.1. MATLAB App Designer kullanılarak geliştirilen Fleet Manager arayüzünün genel görünümü. Arayüzde harita üretimi, robot yerleştirme, planlayıcı seçimi ve simülasyon kontrol bileşenleri ile çoklu robotların başlangıç ve hedef konumları gösterilmektedir.

3. Harita Oluşturma Yöntemleri

3.1 Rastgele Harita Oluşturma (Random Map)

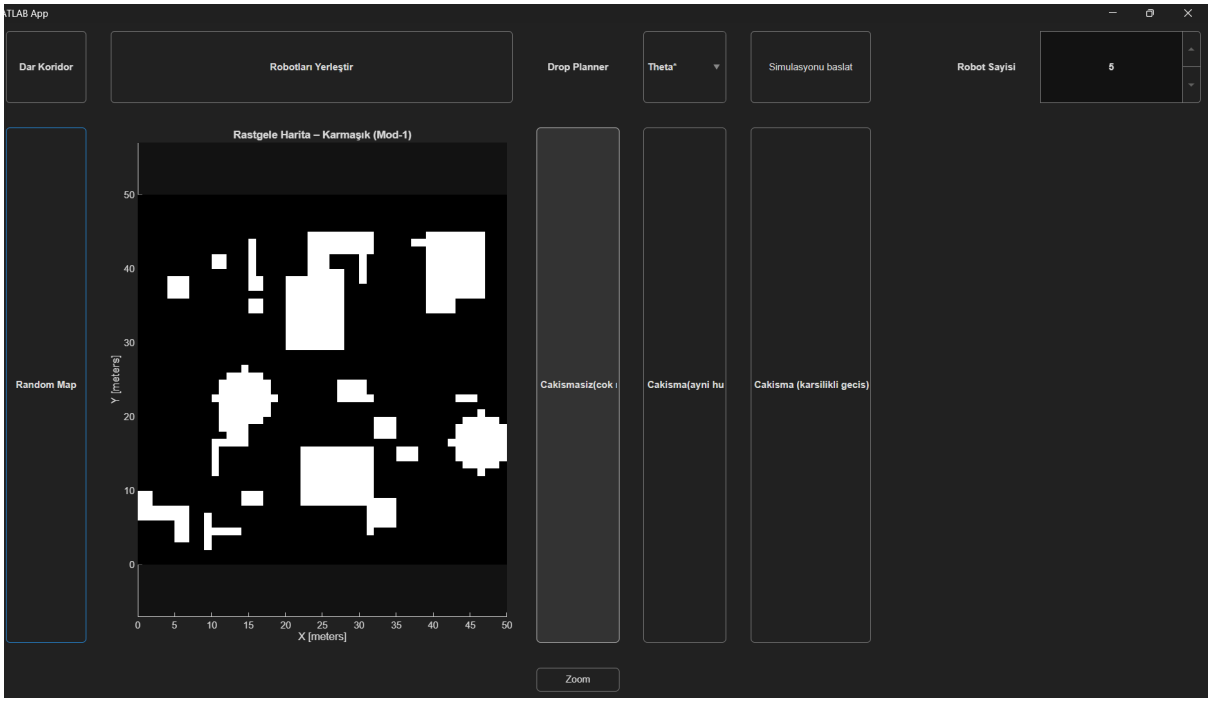
Bu projede kullanılan rastgele harita oluşturma yaklaşımı, hücre bazlı bağımsız rastgelelikten farklı olarak şekil tabanlı ve olasılıksal bir üretim mantığına dayanmaktadır. Amaç, hem gerçekçi hem de çok robotlu planlama senaryolarını zorlayacak karmaşıklıkta haritalar elde etmektir.

Haritalar 50×50 boyutunda bir ızgara üzerinde oluşturulmakta ve engel yoğunluğu doğrudan hücre oranı ile değil, hedeflenen kaplanan alan oranı üzerinden kontrol edilmektedir. Harita üretimi sırasında, toplam doluluk oranı belirlenen eşik değere ulaşana kadar farklı geometrik engel şekilleri haritaya eklenmektedir. Bu yaklaşım, küçük ve büyük engellerin dengeli şekilde dağılmasını sağlamaktadır.

Engel üretiminde birden fazla geometrik primitif kullanılmaktadır. Dikdörtgen bloklar, L-şekilli yapılar, paralel duvarlar ve eliptik engeller farklı olasılıklarla seçilmektedir. Küçük boyutlu engellerin daha yüksek olasılıkla, büyük boyutlu engellerin ise daha düşük olasılıkla üretilmesi sağlanarak haritanın aşırı tıkanması önlenmiştir. Her engel için konum, boyut ve yönelim değerleri rastgele ancak belirlenen aralıklar dahilinde seçilmektedir.

Bu şekil tabanlı yaklaşım sayesinde, harita üzerinde dar geçitler, geniş boşluklar ve düzensiz engel kümeleri doğal bir şekilde oluşmaktadır. Böylece çok robotlu planlama algoritmaları için hem çakışmasız hem de potansiyel çakışma içeren senaryolar üretilebilmektedir.

Şekil 3.1



Şekil 3.1. Şekil tabanlı ve olasılıksal yaklaşım ile üretilmiş rastgele karmaşık harita örneği (Mod-1). Haritada farklı boyut ve geometrilere sahip engeller dengeli şekilde dağılmıştır.

3.2 Dar Koridor Harita Oluşturma

Dar koridor harita modu, çok robotlu sistemlerde çakışma ve deadlock senaryolarını bilinçli olarak tetiklemek amacıyla tasarlanmıştır. Bu harita türünde temel hedef, robotların aynı anda dar bir geçidi kullanmak zorunda kaldığı durumlar oluşturarak, karşılıklı bekleme ve kilitlenme (deadlock) olasılığını artırmaktır.

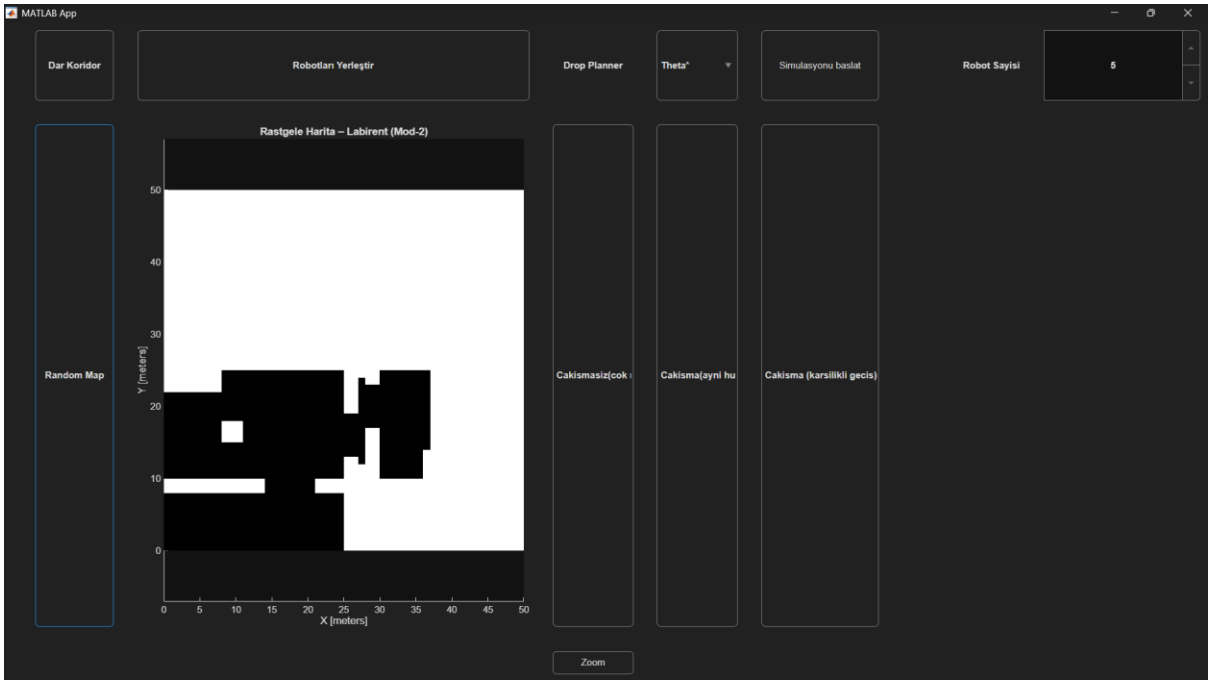
Bu çalışmada dar koridor haritaları, paralel duvar yapıları kullanılarak oluşturulmuştur. Harita üzerinde iki paralel duvar arasında tek veya sınırlı sayıda hücreden oluşan bir geçiş bölgesi tanımlanmıştır. Bu geçiş bölgesi, robotların aynı anda karşılıklı yönlerden ilerlediği senaryolarda doğal bir darboğaz oluşturmaktadır. Duvarların konumu, kalınlığı ve geçit açıklığı rastgele ancak belirli sınırlar içerisinde seçilerek her simülasyonda farklı bir dar koridor geometrisi elde edilmiştir.

Dar koridor senaryoları yalnızca tek tip bir yapıdan oluşmamaktadır. Ana geçit bölgesine ek olarak, koridor içerisine küçük ölçekli engeller ve kısa duvar parçaları yerleştirilerek robotların manevra alanı kısıtlanmıştır. Bu yaklaşım, robotların alternatif kaçış yolları bulmasını zorlaştırarak çakışma ve bekleme ilişkilerinin daha belirgin hale gelmesini sağlamaktadır.

Bu harita modu, özellikle karşılıklı geçiş (head-on passing) senaryoları için kritik öneme sahiptir. İki veya daha fazla robotun dar bir geçitte birbirine doğru hareket etmesi durumunda, robotlar karşılıklı olarak birbirlerini beklemekte ve bu durum wait-for ilişkileri oluşturmaktadır. Bu ilişkiler, simülasyon sırasında görsel olarak oklarla gösterilen wait-for grafiği üzerinden izlenebilmektedir.

Sonuç olarak dar koridor haritaları, sistemin çakışma algılama, deadlock tespiti ve çözüm stratejilerinin etkinliğini değerlendirmek için kontrollü ve tekrarlanabilir bir test ortamı sunmaktadır. Bu mod sayesinde, farklı planlayıcılar ve çakışma yönetim yaklaşımları aynı zor senaryo üzerinde karşılaştırılabilmektedir.

Şekil 3.2



Şekil 3.2. Dar koridor haritası örneği. Paralel duvarlar ve sınırlı geçiş alanı, robotlar arasında karşılıklı geçiş ve deadlock senaryolarının oluşmasını teşvik etmektedir.

3.3 Harita Geçerlilik ve Kalite Kontrolleri

Rastgele ve şekil tabanlı yöntemlerle üretilen haritalar, doğaları gereği her zaman çok robotlu planlama için uygun olmayabilir. Bu nedenle, oluşturulan her harita simülasyona alınmadan önce bir dizi geçerlilik ve kalite kontrolünden geçirilmektedir. Kontrolleri geçemeyen haritalar otomatik olarak elenmekte ve yeniden üretilmektedir.

İlk olarak, harita üzerinde en az bir robot–hedef çifti arasında geçerli bir yol bulunup bulunmadığı kontrol edilmektedir. Bu amaçla, haritadaki serbest hücreler arasından rastgele seçilen noktalar için A* algoritması kullanılarak yol denemeleri yapılmaktadır. Eğer belirlenen denemeler sonucunda en az bir geçerli yol bulunamazsa, harita geçersiz kabul edilmekte ve yeniden oluşturulmaktadır. Bu kontrol, haritanın tamamen tıkanmış veya işlevsiz olmasını önlemektedir.

İkinci aşamada, tamamen izole edilmiş serbest alan bölgeleri tespit edilmektedir. Bu tür bölgeler, robotların hiçbir şekilde erişemeyeceği kapalı cepler oluşturmakta ve planlama algoritmalarının başarımını olumsuz etkilemektedir. Serbest hücreler üzerinde bağlı bileşen analizi (connected components) uygulanarak en büyük erişilebilir bölge belirlenmekte, bunun dışında kalan izole serbest alanlar otomatik olarak engel olarak işaretlenmektedir. Bu işlem, haritanın tek ve bütüncül bir çalışma alanı sunmasını sağlamaktadır.

Üçüncü kontrol aşamasında, çok kısa ve anlamsız çıkmaz sokaklar ele alınmaktadır. Uzunluğu yalnızca 1–2 hücre olan bu tür çıkmazlar, robotların gereksiz yere bu bölgelere girip takılmasına neden olabilmektedir. Bu nedenle, az sayıda serbest komşuya sahip hücreler belirli bir olasılık dahilinde temizlenmekte veya engel haline getirilmektedir. Olasılıksal yaklaşım sayesinde, haritanın

tamamen steril hale gelmesi engellenirken, planlama açısından anlamsız yapıların büyük ölçüde azaltılması sağlanmaktadır.

Bu kalite kontrol adımları, hem rastgele karmaşık haritalar hem de labirent ve dar koridor modları için uygulanmaktadır. Sonuç olarak, simülasyona alınan her harita; erişilebilir, anlamlı ve çok robotlu planlama açısından zorluk içeren bir yapıya sahip olmaktadır. Bu yaklaşım, simülasyon sonuçlarının tutarlı ve tekrarlanabilir olmasını sağlamaktadır.

4. Çok Robotlu Yapı

Geliştirilen Fleet Manager sistemi, aynı harita üzerinde birden fazla robotun eş zamanlı olarak hareket edebileceği şekilde tasarlanmıştır. Simülasyonda kullanılan robot sayısı kullanıcı tarafından belirlenebilmekte olup, bu çalışmada robot sayısı öğrenci numarasına bağlı değişken bir değer veya sabit olarak 10 robot olacak şekilde ayarlanabilmektedir.

Her robot için başlangıç ve hedef konumları birbirinden farklı olacak şekilde tanımlanmıştır. Robotların ve hedeflerin engel hücreleri üzerinde veya birbirleriyle çakışacak biçimde yerleştirilmesine izin verilmemektedir. Bu sayede simülasyonun başlangıç koşulları tutarlı ve anlamlı bir şekilde oluşturulmaktadır.

Robotlar arayüz üzerinde farklı renkler veya işaretler kullanılarak görselleştirilmektedir. Böylece simülasyon sırasında robotların konumları, hedefleri ve izledikleri yollar kolaylıkla ayırt edilebilmektedir. Bu görsel ayrım, özellikle çakışma ve deadlock senaryolarının analiz edilmesi sırasında önemli bir kolaylık sağlamaktadır.

4.1 Robot–Hedef Mesafe Sınıfları

Robot ve hedef yerleşimleri kontrolsüz rastgelelik ile gerçekleştirilmemektedir. Her robot için hedef seçimi, robot ile hedef arasındaki en kısa yol uzunluğuna dayalı mesafe sınıfları kullanılarak yapılmaktadır. Bu yaklaşım, simülasyonlarda farklı zorluk seviyelerine sahip senaryoların sistematik olarak üretilmesini sağlamaktadır.

Mesafe sınıfları, haritanın boyutları dikkate alınarak nicel olarak tanımlanmıştır. Robot ile hedef arasındaki en kısa yol uzunluğunun harita çapına oranı kullanılarak üç temel sınıf oluşturulmuştur. Yakın hedefler, harita çapının yaklaşık %10–%20'si aralığında; orta mesafe hedefler %30–%50 aralığında; uzak hedefler ise %60 ve üzerindeki mesafelerde tanımlanmıştır.

Her robot için hedef, ilgili mesafe sınıfını sağlayan serbest hücreler arasından rastgele seçilmektedir. Engel üzerinde bulunan, erişilemeyen veya başka bir robotun hedefi ile çakışan hücreler aday kümeden çıkarılmaktadır. Bu sayede hem rastgelelik korunmakta hem de senaryo üretimi kontrol altında tutulmaktadır.

Bu mesafe tabanlı hedef seçimi, kısa ve basit senaryolardan uzun ve karmaşık senaryolara kadar geniş bir test yelpazesi sunmaktadır. Özellikle uzak hedef senaryolarında robotlar arası etkileşim ve çakışma olasılığı belirgin şekilde artmaktadır.

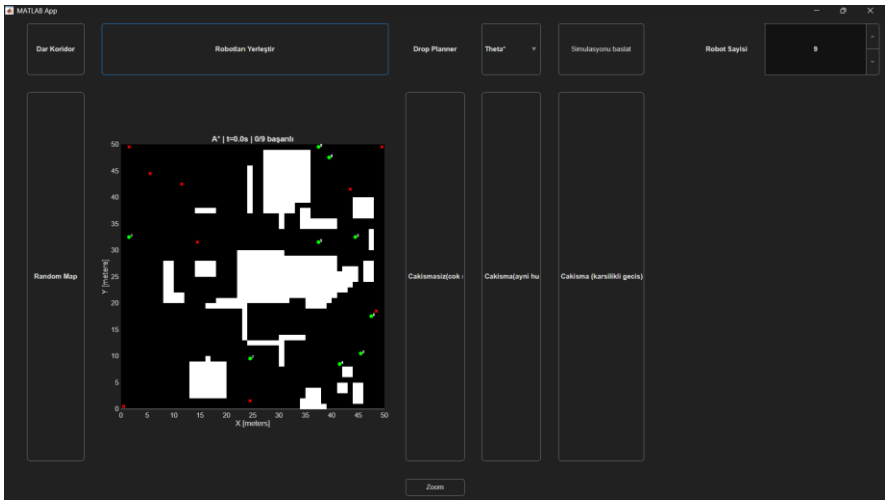
4.2 Senaryo Dağılımı ve Rastgelelik Kontrolü

Simülasyonlar tek tip senaryolar üzerinden yürütülmemektedir. Sistem, farklı zorluk seviyelerini temsil eden senaryoları otomatik olarak üretmektedir. Bu senaryolar; orta mesafeli ve düşük çakışma olasılığına sahip durumlar, uzak mesafeli ve yüksek çakışma olasılığı içeren durumlar ile dar koridorlarda karşılıklı geçiş ve deadlock senaryolarını kapsamaktadır.

Her simülasyon çalışmasında kullanılan rastgele tohum (random seed) değeri kaydedilmektedir. Bu sayede aynı harita ve senaryo koşulları yeniden üretilebilmekte ve farklı planlayıcılar veya çakışma yönetim stratejileri adil bir şekilde karşılaştırılabilmektedir. Rastgelelik, tamamen serbest bırakılmak yerine senaryo çeşitliliğini artıracak şekilde kontrol altında tutulmaktadır.

Bu yaklaşım sayesinde, geliştirilen sistem yalnızca tek bir duruma özel değil, çok farklı ve zorlayıcı koşullar altında test edilebilen esnek bir simülasyon altyapısı sunmaktadır.

Şekil 3.2



Şekil 4.1. Aynı harita üzerinde farklı başlangıç ve hedef noktalarına sahip çoklu robotların yerleşimi. Her robotun başlangıç (yeşil) ve hedef (kırmızı) konumları birbirinden farklı olacak şekilde belirlenmiştir.

5. Global Yol Planlama Yöntemleri

5.1 Kullanılan Yol Planlayıcılar

Bu çalışmada, çoklu robotların statik engeller içeren ızgara haritalarda hedeflerine ulaşabilmesi için üç farklı global yol planlama algoritması kullanılmıştır:

- **A***
- **PRM (Probabilistic Roadmap)**
- **Theta***

Planlayıcılar, kullanıcı arayüzü üzerinden seçilebilmekte ve aynı senaryo farklı algoritmalarla karşılaştırmalı olarak çalıştırılabilmektedir.

5.1.1 A* Algoritması

A* algoritması, sezgisel (heuristic) fonksiyon kullanan en yaygın ızgara tabanlı yol planlama yöntemlerinden biridir. Bu çalışmada A*, robotun başlangıç ve hedef hücreleri arasındaki en düşük maliyetli yolu bulmak amacıyla kullanılmıştır.

Sezgisel fonksiyon olarak Öklidyen mesafe tercih edilmiştir. Algoritma, engel hücrelerini dikkate alarak 8-komşuluk (diagonal hareket dahil) yapıda çalışmaktadır. Köşe kesme (corner cutting) durumları engellenerek fiziksel olarak geçersiz yolların oluşması önlenmiştir.

A* algoritması, deterministik yapısı sayesinde güvenilir ve tekrar edilebilir sonuçlar üretmektedir. Ancak yolun yalnızca hücre merkezlerinden geçmesi nedeniyle bazı senaryolarda gereğinden uzun veya köşeli yollar üretebilmektedir.

5.1.2 PRM (Probabilistic Roadmap)

PRM algoritması, harita üzerinde rastgele örneklenen düğümlerden oluşan bir grafik yapı kurarak yol planlaması yapmaktadır. Bu yaklaşım özellikle geniş ve karmaşık ortamlarda esnek çözümler sunabilmektedir.

Bu projede PRM, yoğun engel içeren haritalarda çalışabilecek şekilde düğüm sayısı ve bağlantı mesafesi ayarlanarak kullanılmıştır. Başlangıç ve hedef noktaları, oluşturulan yol haritasına bağlanarak en uygun yol hesaplanmıştır.

PRM'nin rastgele örnekleme yapısı nedeniyle her çalıştırmada aynı sonucu üretmemesi mümkündür. Bu durum, algoritmanın hem güçlü hem de zayıf yönü olarak değerlendirilmektedir.

5.1.3 Theta* Algoritması

Theta*, A* algoritmasının geliştirilmiş bir versiyonu olup, hücre merkezlerine bağlı kalmadan görüş hattı (line-of-sight) kontrolü ile daha düzgün ve kısa yollar üretmeyi amaçlamaktadır.

Bu çalışmada Theta*, A*'ya ek olarak hücreler arasında doğrudan görüş olup olmadığını kontrol ederek köşeleri kesen daha doğal yollar oluşturmuştur. Böylece robotların daha az yön değiştirerek ilerlemesi sağlanmıştır.

Theta* algoritması, özellikle açık alanlarda ve karmaşık engel geometrilerinde A*'ya kıyasla daha akıcı yollar üretmiştir.

6. Simülasyon ve Deneysel Sonuçlar

Bu bölümde geliştirilen sistemin, farklı senaryolar ve planlayıcılar altında nasıl davrandığı incelenmiştir. Simülasyonlar MATLAB App Designer ortamında gerçek zamanlı olarak gerçekleştirilmiş, robotların hareketleri, çarpışma durumları ve deadlock senaryoları gözlemlenmiştir.

6.1 Simülasyon Ortamı ve Parametreler

Simülasyonlar, 50×50 boyutlarında ikili işgal (binary occupancy) haritaları üzerinde gerçekleştirilmiştir. Robotlar diferansiyel sürüş modeline sahip olup sabit zaman adımı (dt = 0.1 s) ile hareket etmektedir.

Robotların maksimum doğrusal hızı sabit tutulmuş, her robot kendisine atanmış başlangıç ve hedef noktaları arasında global yol planlayıcılar (A*, PRM, Theta*) kullanılarak yönlendirilmiştir.

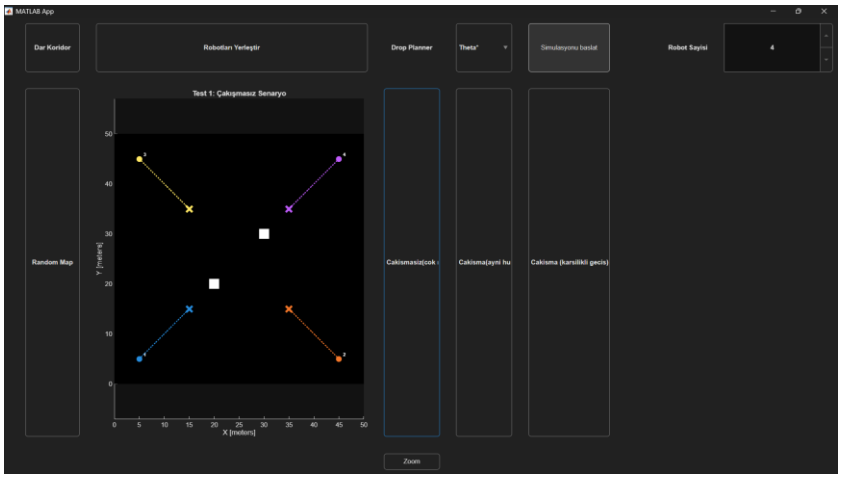
Simülasyon süresi üst sınırı 200 saniye olarak belirlenmiş ve hedefe ulaşan robotlar tamamlanmış (finişe) olarak işaretlenmiştir.

6.2 Test Senaryoları

6.2.1 Çakışmasız Çok Robot Senaryosu

Bu senaryoda robotlar, birbirlerinin yolları ile kesişmeyecek şekilde farklı bölgelere yerleştirilmiştir. Amaç, çarpışma veya deadlock durumu oluşmadan çoklu robotların hedeflerine ulaşabildiğini göstermektir.

Şekil 6.2.1

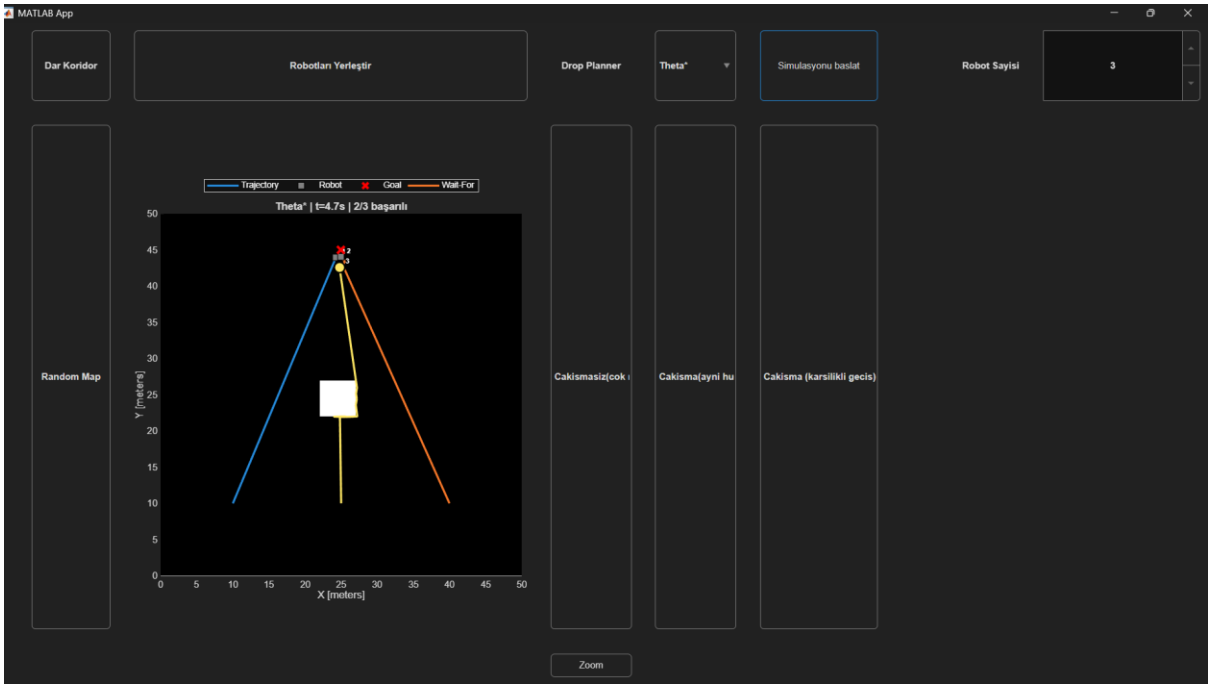


6.2.2 Aynı Hücre / Ortak Hedef Senaryosu

Bu senaryoda birden fazla robot, aynı hedef hücreye yönlendirilmiştir. Amaç, robotların aynı hücreye eşzamanlı olarak ulaşmaya çalışması durumunda oluşabilecek çakışmaların sistem tarafından algılanıp algılanmadığını test etmektir.

Simülasyon sırasında robotlar hedefe yaklaştıklarında, çakışma algılama mekanizması devreye girmiş ve aynı anda aynı hücreye ilerlemek isteyen robotlar bekleme durumuna alınmıştır. Böylece robotların fiziksel olarak çarpışması engellenmiştir.

Şekil 6.2.2



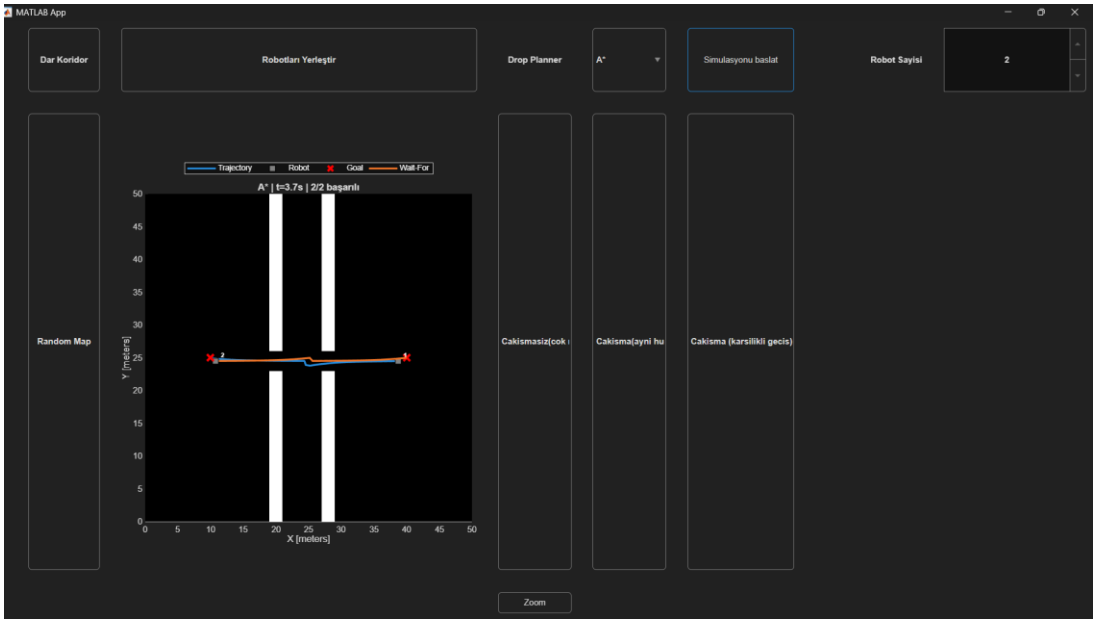
Şekil 6.2.2 Aynı hedef hücreye yönlendirilen robotların simülasyon sonunda oluşturduğu bekleme ve etkileşim durumu.

6.2.3 Karşılıklı Geçiş (Deadlock Potansiyeli)

Bu senaryoda her ne kadar dar bir geçitte karşılıklı ilerleme durumu oluşturulmuş olsa da, sistemin yerel kaçınma ve öncelik mekanizmaları sayesinde robotlar birbirlerini bloke etmeden ilerleyebilmiştir. Bu durum, karşılıklı geçiş senaryosunun her zaman deadlock ile sonuçlanmadığını; uygun kontrol stratejileri ile potansiyel deadlock durumlarının simülasyon aşamasında önlenebildiğini göstermektedir.

Şekil 6.2.3

Şekil 6.3 Dar koridorda karşılıklı yönde ilerleyen robotların, yerel kaçınma mekanizmaları sayesinde deadlock oluşmadan geçiş yapabildiği durum.



Şekil 6.6'da robotların dar bir geçitte karşılıklı ilerlemelerine rağmen, sistemin çakışma yönetimi sayesinde karşılıklı bloke olmadan yollarına devam edebildikleri görülmektedir.

6.3 Deadlock Tespiti ve Çözüm Stratejileri

Bu bölümün amacı, çoklu robot sisteminde oluşabilecek deadlock durumlarının **nasıl algılandığını** ve **hangi stratejilerle çözüldüğünü** açıklamaktır.

6.3.1 Deadlock Tespiti

Sistem içerisinde deadlock durumu, bir veya daha fazla robotun ilerleyemediği ve karşılıklı olarak birbirini beklediği durumlar üzerinden tespit edilmektedir. Bu amaçla her robot için anlık olarak beklediği robotlar izlenmekte ve bekleme ilişkileri kullanılarak bir *wait-for graph* yapısı oluşturulmaktadır.

Eğer bu grafik üzerinde döngüsel bir bekleme ilişkisi (circular wait) tespit edilirse, sistem bu durumu deadlock olarak değerlendirmektedir. Bunun yanı sıra, belirli bir süre boyunca ilerleme kaydedemeyen birden fazla robotun bulunması da deadlock potansiyeli olarak ele alınmaktadır.

6.3.2 Deadlock Türleri

Uygulamada deadlock durumları üç farklı başlık altında ele alınmıştır.

Döngüsel Bekleme (Circular Wait): Robotların birbirlerini zincirleme şekilde beklemesi sonucu oluşan klasik deadlock durumudur.

Uzun Süreli Takılma (Long Stuck): Robotların belirli bir süre boyunca ilerleme kaydedememesi durumunda oluşan deadlock türüdür.

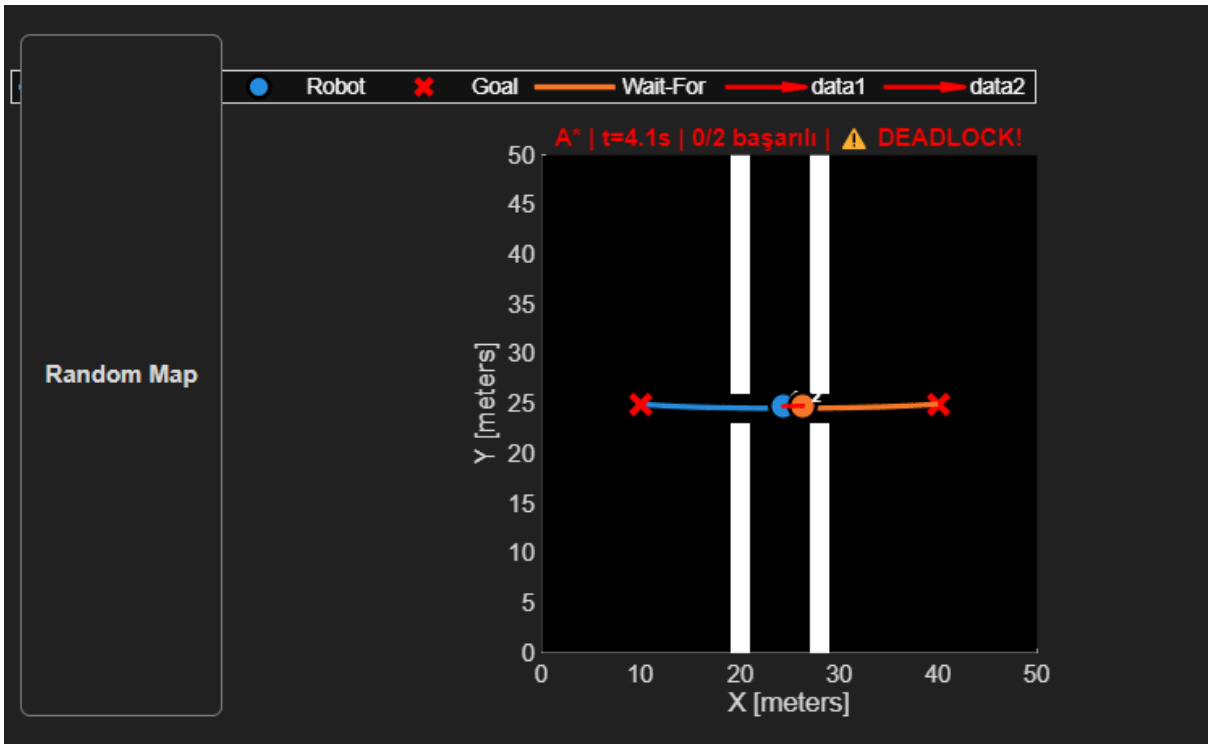
Mekânsal Deadlock (Spatial Deadlock): Robotların birbirlerine çok yakın konumlarda bulunmasına rağmen ilerleyememeleri durumunda ortaya çıkan deadlock türüdür.

6.3.3 Deadlock Çözüm Stratejileri

Deadlock tespit edildiğinde sistem, çözüm için çok katmanlı bir strateji uygulamaktadır. Öncelikle robotlar arasında bir öncelik sistemi tanımlanmış ve belirlenen robotun geri çekilmesi sağlanmıştır.

Eğer bu yöntem yeterli olmazsa, robotlara rastgele yan hareketler uygulanarak kilitlenmiş durumdan çıkmaları hedeflenmiştir. Son çare olarak ise, uzun süre ilerleyemeyen robotlar için yol yeniden planlama mekanizması devreye sokulmuştur. Bu çok aşamalı yaklaşım sayesinde deadlock durumları simülasyon sırasında dinamik olarak çözülebilmektedir.

Şekil 6.3.3. Deadlock anında robotlar arasındaki wait-for ilişkilerinin görselleştirilmesi.



7. Simülasyon Sonuçları ve Performans Analizi

Bu bölümde farklı senaryolar ve yol planlayıcılar altında gerçekleştirilen simülasyonlardan elde edilen sonuçlar analiz edilmiştir. Değerlendirme, robotların hedefe ulaşma başarısı, simülasyon süresi, çakışma sayısı ve deadlock oluşumu gibi metrikler üzerinden yapılmıştır.

7.1 Değerlendirme Kriterleri

Performans analizi aşağıdaki ölçütler kullanılarak gerçekleştirilmiştir:

- **Başarı Oranı:** Hedefe ulaşan robot sayısının toplam robot sayısına oranı
- **Simülasyon Süresi:** Tüm robotların hedefe ulaşması veya simülasyonun sonlanmasına kadar geçen süre
- **Çakışma Sayısı:** Simülasyon boyunca algılanan çakışma durumlarının sayısı
- **Deadlock Sayısı:** Tespit edilen deadlock veya deadlock potansiyeli durumlarının sayısı

Bu metrikler, farklı planlayıcıların ve senaryoların karşılaştırılmasını mümkün kılmaktadır.

7.2 Planlayıcı Performans Karşılaştırması

Senaryo	**Planlayıcı**	**Başarılı Robot**	**Başarı Oranı (%)**	**Simülasyon Süresi (s)**	**Deadlock**
Mod-1 (Karmaşık)	Theta*	7	%78	40	Var (2 robot)
Mod-1 (Karmaşık)	A*	5	%56	9	Yok
Mod-1 (Karmaşık)	PRM	4	%44	9	Var (4 robot)
Mod-2 (Labirent)	Theta*	5	%56	15	Yok
Mod-2 (Labirent)	A*	3	%33	10	Var (2 robot)
Mod-2 (Labirent)	PRM	5	%56	20	Yok
Dar Koridor	Theta*	6	%67	16	Yok
Dar Koridor	PRM	2	%22	10	Yok
Dar Koridor	A*	6	%67	10	Yok

Tablo 7.1'de görüldüğü üzere, planlayıcıların performansı senaryo türüne bağlı olarak önemli farklılıklar göstermektedir. Karmaşık harita (Mod-1) senaryosunda A* algoritması deadlock oluşturmadan hızlı sonuç üretmiş ancak başarı oranı sınırlı kalmıştır. Theta* algoritması daha yüksek başarı oranı sağlasa da bazı durumlarda deadlock çözülememiştir.

Labirent tipi haritalarda (Mod-2) PRM ve Theta* algoritmaları daha dengeli performans sergilerken, A* algoritması bazı robotlar için deadlock durumuna girmiştir. Dar koridor senaryosunda ise tüm planlayıcılar deadlock oluşturmadan çalışmış, ancak A* algoritmasının başarı oranı belirgin şekilde düşük kalmıştır.

7.3 Senaryolara Göre Gözlemsel Analiz

Çakışmasız senaryolarda tüm planlayıcılar yüksek başarı oranı göstermiştir. Dar koridor ve karşılıklı geçiş senaryolarında ise çakışma ve deadlock potansiyeli belirgin şekilde artmıştır. Bu senaryolarda sistemin çakışma yönetimi ve deadlock çözüm stratejileri devreye girerek simülasyonun başarılı şekilde tamamlanmasını sağlamıştır.

7.4 Rastgelelik ve Tekrarlanabilirlik

Her simülasyon çalışmasında kullanılan rastgele tohum (random seed) değeri sistem tarafından kaydedilmiştir. Bu sayede aynı senaryoların tekrar üretilebilmesi ve sonuçların karşılaştırılabilir olması sağlanmıştır. Rastgeleliğin kontrollü şekilde kullanılması, sistemin hem esnek hem de tekrarlanabilir olmasını mümkün kılmıştır.

8. Hız ve Tekerlek Analizi

Bu bölümde simülasyon sırasında robotların adım adım hareketlerinden türetilen **doğrusal hız**, **açısal hız** ve **tekerlek hızları** analiz edilmiştir. Hesaplamalarda **diferansiyel sürüş (differential drive)** kinematik modeli kullanılmıştır.

Not: Analiz, temsil edici olması amacıyla simülasyonu başarıyla tamamlayan **tek bir robot** üzerinden yapılmıştır.

8.1 Doğrusal ve Açısal Hız Analizi

Robotun ardışık zaman adımlarındaki konum değişimleri kullanılarak doğrusal hız $v(t)$ve yönelim değişimi kullanılarak açısal hız $\omega(t)$hesaplanmıştır.

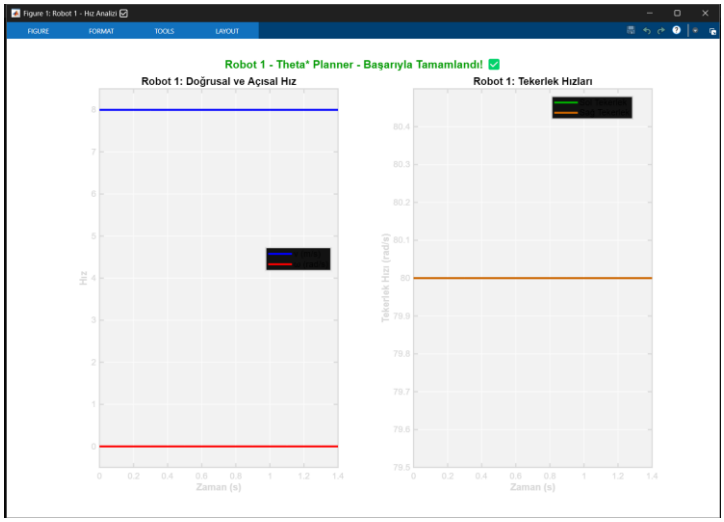
- **Doğrusal hız (v):**

$$v(t) = \frac{\| p(t) - p(t-\Delta t) \|}{\Delta t}$$

- **Açısal hız (ω):**

$$\omega(t) = \frac{\theta(t) - \theta(t-\Delta t)}{\Delta t}$$

Şekil 8.1. Theta* + Mod-1 senaryosunda seçilen temsil edici bir robot için doğrusal/açısal hızlar ve sol-sağ tekerlek hızları.



Şekil 8.1'de görüldüğü üzere robotun doğrusal hızı simülasyon boyunca kararlı seyretmiş, belirgin yön değişimi gerektirmeyen bu koşuda açısal hız sıfıra yakın kalmıştır. Buna bağlı olarak sol ve sağ tekerlek hızlarının eşit olduğu gözlemlenmektedir. Bu durum, diferansiyel sürüş modelinin simülasyon içerisinde tutarlı şekilde uygulandığını göstermektedir.

8.3 Genel Değerlendirme

Elde edilen hız profilleri, robotun hem düz ilerleme hem de manevra gerektiren durumlarda fiziksel olarak tutarlı davranışlar sergilediğini göstermektedir. Doğrusal ve açısal hızlardan türetilen tekerlek hızları, simülasyonun kinematik açıdan gerçekçi olduğunu ortaya koymaktadır.

9. Sonuç ve Değerlendirme

Bu çalışmada, çoklu robotların aynı ızgara tabanlı harita üzerinde güvenli ve etkin şekilde hareket edebilmesini sağlayan bir **Fleet Manager sistemi**, MATLAB App Designer kullanılarak geliştirilmiştir. Sistem; rastgele ve dar koridor içeren haritalar üzerinde global yol planlama, çakışma yönetimi, deadlock tespiti ve simülasyon fonksiyonlarını bütünleşik olarak sunmaktadır.

Geliştirilen uygulama kapsamında **A***, **PRM** ve **Theta*** olmak üzere üç farklı global yol planlama algoritması kullanıcı tarafından seçilebilir hale getirilmiş ve farklı senaryolarda karşılaştırmalı olarak değerlendirilmiştir. Deneysel sonuçlar, planlayıcı performanslarının harita yapısına ve robot yoğunluğuna bağlı olarak önemli ölçüde değiştiğini göstermiştir. Özellikle Theta* algoritması, karmaşık ve açık alan içeren haritalarda daha düzgün ve kısa yollar üretirken; A* algoritması deterministik yapısı sayesinde bazı senaryolarda daha hızlı sonuçlar sağlamıştır. PRM algoritması ise rastgele örnekleme yaklaşımı nedeniyle belirli durumlarda daha değişken bir performans sergilemiştir.

Çoklu robot etkileşimlerinin yoğun olduğu senaryolarda çakışma ve deadlock potansiyeli belirgin hale gelmiştir. Bu tür durumlarda sistem, **wait-for graph** tabanlı deadlock tespiti ve çok aşamalı çözüm stratejileri ile kilitlenme durumlarını algılayabilmiş ve çoğu senaryoda simülasyonun başarıyla tamamlanmasını sağlamıştır. Dar koridor senaryoları, geliştirilen çakışma yönetimi mekanizmalarının etkinliğini değerlendirmek açısından özellikle kritik bir test ortamı sunmuştur.

Ayrıca robotların doğrusal, açısal ve tekerlek hızları analiz edilerek diferansiyel sürüş modelinin simülasyon içerisinde tutarlı şekilde uygulandığı gösterilmiştir. Hız profilleri, robotların çevresel etkileşimlere ve manevra gereksinimlerine uygun şekilde tepki verdiğini ortaya koymuştur.

Sonuç olarak geliştirilen Fleet Manager uygulaması, çoklu robot sistemleri için temel planlama, çakışma yönetimi ve analiz ihtiyaçlarını karşılayan, genişletilebilir bir simülasyon altyapısı sunmaktadır. Gelecek çalışmalar kapsamında, dinamik engellerin sisteme dahil edilmesi, öncelik tabanlı daha gelişmiş deadlock çözüm stratejileri ve gerçek robot platformları ile entegrasyon gibi geliştirmeler yapılabilir.

KAYNAKLAR

1. **MathWorks.**
   Robotics System Toolbox Documentation.
   https://www.mathworks.com/help/robotics/

2. **MathWorks.**
   Image Processing Toolbox Documentation.
   https://www.mathworks.com/help/images/

3. LaValle, S. M. (2006).
Planning Algorithms.
Cambridge University Press.
(A*, PRM ve yol planlama altyapısı için temel referans)

4. Nash, A., Koenig, S., & Tovey, C. (2007).
Theta: Any-Angle Path Planning on Grids.*
AAAI Conference on Artificial Intelligence.

5. Corke, P. (2011).
Robotics, Vision and Control.
Springer.
(Diferansiyel sürüş, tekerlek hızları ve robot kinematiği için