

FLEET MANAGER

Çoklu Robot Yol Planlama ve Çatışma Yönetimi

MATLAB App Designer Tabanlı Simülasyon Projesi

3 Harita Tipi

Random, Koridor, Maze

4 Planlayıcı

A*, CBS, WHCA*, Prioritized

4 Çatışma Stratejisi

Reactive, Wait, Token, Reservation

Gerçek Zamanlı Analiz

Hız grafikleri, Metrikler

Proje: **Çoklu Robot Filo Yönetimi**

Hazırlayan: Ömer Faruk Özdemir

Numara: 240201122

Tarih: 3 Ocak 2026



İÇİNDEKİLER

1	Devriye ve Güvenlik Senaryosu	3
2	Sistem Mimarisi	4
3	Harita Oluşturma	6
4	Yol Planlama Algoritmaları	7
5	Çalışma ve Deadlock Yönetimi	9
6	Robot Kinematiği	10
7	Test Senaryoları	12
8	Sonuç ve Değerlendirme	13

⌚ Proje Tanımı

Fleet Manager, çoklu robot koordinasyonu için geliştirilmiş kapsamlı bir simülasyon platformudur. Sistem, MATLAB App Designer ile tasarlanmış olup, gerçek zamanlı yol planlama, çatışma tespiti ve çözümü, deadlock yönetimi gibi kritik fonksiyonları içermektedir.

10 adet

Robot

4+

Algoritma

360°

Tam kapsam

1.5 s

Saniye/Adım

Operasyon Ortamı

Sistem, 2D ızgara tabanlı (occupancy grid) bir ortamda çalışmaktadır. Robotlar, binaryOccupancyMap üzerinde hareket eder ve hedeflerine ulaşmak için planlı yolları takip eder.

Parametre	Değer / Açıklama
gMaps Harita Boyutu	Kullanıcı tarafından ayarlanabilir (MapSizeSpinner)
♀ Çözünürlük	1 hücre = 1 birim (resolution = 1)
🤖 Robot Sayısı	SeedField_2 Spinner ile dinamik seçim
⌚ Hedef Mesafesi	Yakın (0.1-0.2), Orta (0.3-0.5), Uzak (0.6-1.0) köşegen
🎲 Seed Değeri	Tekrarlanabilirlik için rastgele tohum

İstenen Güvenlik Riskleri

Risk Kategorisi	Açıklama	Çözüm Stratejisi
⚠ Hücre Çatışması	İki robot aynı hücreye gitmeye çalışır	Wait & Priority
☒ Swap Çatışması	İki robot yer değiştirmeye çalışır	Token Passing
🔒 Deadlock	Döngüsel bekleme durumu	Wait-For Graph analizi
♀ Hedef Blokajı	Robot hedefinde başka robot var	Reactive Re-planning

Fleet Manager, MATLAB App Designer'ın nesne yönelimli yapısını kullanarak modüler bir mimaride tasarlanmıştır. Arayüz bileşenleri, veri yapıları ve algoritmalar ayrı metodlar halinde organize edilmiştir.

Arayüz Bileşenleri

Bileşen	Tip	İşlev
MapAxes	UIAxes	Harita görselleştirme alanı
BtnRandomMap	Button	Harita oluşturma (tip seçimine göre)
MapTypeDropdown	DropDown	Harita tipi seçimi (Rastgele/Koridor/Labirent)
BtnPlaceRobots	Button	Robotları yerleştirme
GlobalPlannerDrop	DropDown	Planlayıcı seçimi (4 seçenek)
ConflictDrop	DropDown	Çatışma stratejisi seçimi
HeuristicDrop	DropDown	Heuristik seçimi
PrimaryButton	Button	Simülasyon başlat/durdur
DensitySlider	Slider	Engel yoğunluğu ayarı
SeedField_2	Spinner	Robot sayısı seçimi
InfoBox	TextArea	Dinamik bilgi mesajları

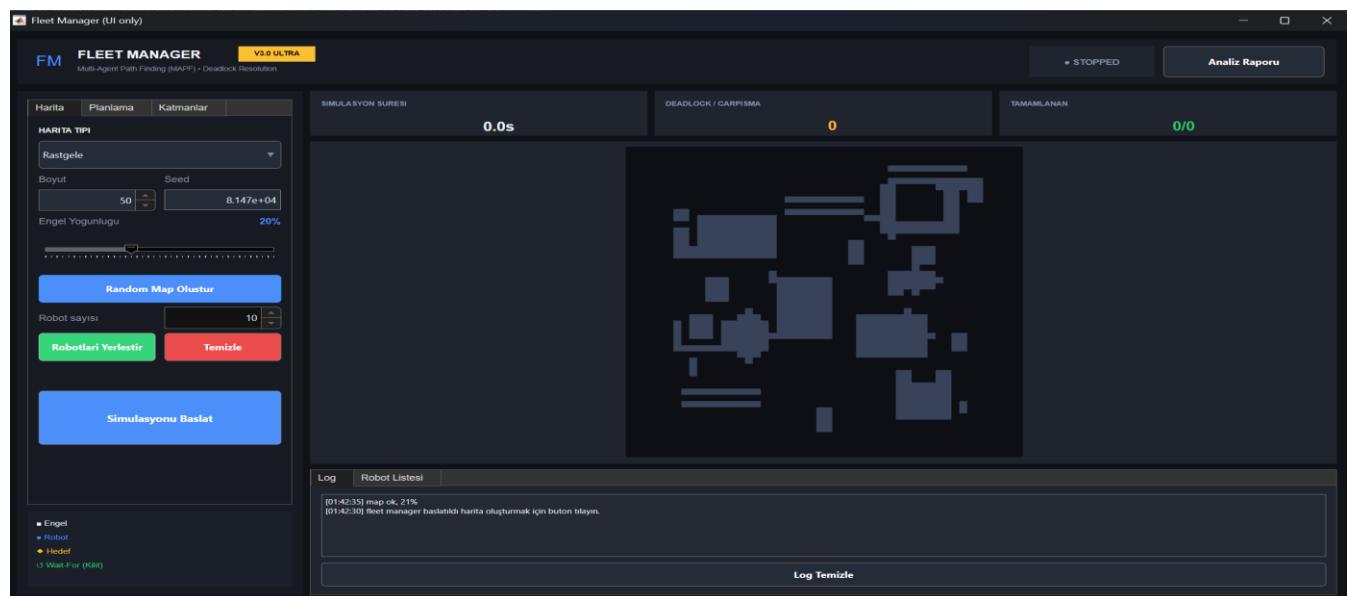
Robot Veri Yapısı (Struct)

Robot Struct Alanları

id: Robot kimlik numarası (1-N)
start: Başlangıç pozisyonu [x, y]
goal: Hedef pozisyonu [x, y]
currentPos: Anlık konum [x, y]
path: Aktif yol segmenti (Nx2 matris)
fullPath: Tam planlanan yol (WHCA* için)
pathIndex: Yol üzerindeki mevcut indeks
history: Geçmiş pozisyonlar
status: Durum (hareket/bek/tamamlandı/yol yok)
waitingFor: Beklenen robot ID
waitCount: Bekleme sayacı

Uygulama Yapısı

Katman	Fonksiyonlar	Sorumluluk
UI Layer	Callbacks, Event Handlers	Kullanıcı etkileşimi
Logic Layer	BtnPlaceRobotsButtonPushed, runSimulation	İş mantığı
Planning Layer	gridAStar, planCBS, spaceTimeAStar, planPrioritizedSpaceTime	Yol planlama
Conflict Layer	checkConflict, detectCyclicDeadlock	Çatışma yönetimi
Visualization	renderMap, drawWaitForGraph, updateRobotTable	Görselleştirme



3 HARİTA OLUŞTURMA

Sistem, üç farklı harita tipi desteklemektedir. Her harita tipi, farklı senaryoları test etmek için optimize edilmiştir.



Rastgele



Dar Koridor



Labirent



Doğrulama

Rastgele Harita (createRandomMap)

Şekil tabanlı rastgele harita üretimi, geometrik primitifler kullanarak doğal görünümlü ortamlar oluşturur. Hedef engel yoğunluğuna ulaşana kadar şekiller eklenir.

Engel Primitifi	Olasılık	Boyut Aralığı	Görsel
Küçük Dikdörtgen	%25	2-4 hücre	■
Büyük Dikdörtgen	%15	5-10 hücre	■■■■
L-Şekilli	%20	4-8 hücre	■■■
U-Şekilli	%12	5-10 hücre	■■■
Dar Geçit	%18	6-14 hücre	
Eliptik	%10	2-5 yarıçap	○

Dar Koridor Haritası (createCorridorMap)

Koridor Haritası Özellikleri

- Ortada 2 hücre genişliğinde ana yatay koridor
- Sol ve sağ tarafta geniş odalar (robot toplanma alanları)
- Rastgele sayıda darboğaz noktaları (2-5 adet)
- Dikey bağlantı koridorları (1-2 birim genişlik)
- Kritik darboğaz noktaları (deadlock testi için ideal)

Labirent Haritası (createMazeMap)

Labirent Haritası Özellikleri

- DFS (Depth-First Search) tabanlı labirent üretimi
- Başlangıç noktası: (3, 3) koordinatı
- 2 hücrelik adımlarla ilerler (duvar-yol-duvar yapısı)
- Kenarlarda açık spawn alanları
- Rastgele açılan ekstra geçişler (%2.7 oranında)

Harita Geçerlilik Kontrolleri

Fonksiyon	Açıklama	Algoritma
validateMap()	Yeterli boş alan kontrolü	Boş hücre oranı $\geq %40$, bağlantılılık $\geq %80$
clearDeadEnds()	Cıkmaız sokakları temizle	≤ 1 boş komşusu olan hücreleri engele çevir
ensureConnectivity()	İzole bölgeleri kaldır	BFS ile en büyük bileşen dışını engele çevir
getLargestComponent()	En büyük bileşeni bul	BFS flood-fill algoritması

4

YOL PLANLAMA ALGORİTMALARI

Sistem, kullanıcının seçimine göre 4 farklı yol planlama algoritması sunmaktadır. Her algoritma, farklı senaryolarda avantaj sağlar.

A*	CBS	WHCA*	Prior
Optimal	Multi-Agent	Hızlı	Öncelikli

A* (Standard Optimal)

⌚ A* Formülü: $f(n) = g(n) + h(n)$

$g(n)$: Başlangıçtan n düşümüne olan gerçek maliyet

$h(n)$: n düşümünden hedefe tahmini maliyet (heuristic)

Optimallik: $h(n)$ kabul edilebilir ise (gerçek maliyeti aşmaz) A* optimal sonuç verir.

Heuristik	Formül	Kullanım Durumu
Manhattan	$ x_1 - x_2 + y_1 - y_2 $	4-yönlü hareket (varsayılan)
Euclidean	$\sqrt{((x_1 - x_2)^2 + (y_1 - y_2)^2)}$	Düz çizgi mesafesi
Chebyshev	$\max(x_1 - x_2 , y_1 - y_2)$	8-yönlü hareket

CBS (Multi-Agent)

Conflict-Based Search, çoklu robot senaryoları için tasarlanmış bir algoritmadır. Sistemimizde basitleştirilmiş bir versiyon kullanılmaktadır:

- ▶ Önce tüm robotlar bağımsız A* ile planlanır
- ▶ Rezervasyon tablosu ile çakışmalar tespit edilir (x, y, t)
- ▶ Çakışma bulunursa yüksek ID'li robot yeniden planlanır
- ▶ Diğer robotların yolları kısıtlama olarak eklenir
- ▶ Maksimum 10 iterasyon ile çözüm aranır

WHCA* (Windowed Hierarchical A*)

⚡ Pencereli A* Yaklaşımı

Tam yolu hesaplar, ancak sadece ilk N adım uygular (varsayılan: 10).

Her pencere sonunda yeniden planlama yapılır.

fullPath: Tam hesaplanan yol saklanır.

path: Aktif pencere segmenti (1-10 adım).

Avantaj: Dinamik ortamlarda esneklik sağlar.

Prioritized Planning

Önceliklendirilmiş planlama, robotları ID sırasına göre planlar:

- ▶ Küçük ID = Yüksek öncelik
- ▶ Space-Time A* ile zamanlı yol planlama
- ▶ Önce planlanan robotların yolları rezervasyon tablosuna eklenir
- ▶ Sonraki robotlar bu rezervasyonlardan kaçınır
- ▶ Hedefte 5 adım ekstra bekleme rezervasyonu yapılır

5

ÇATIŞMA VE DEADLOCK YÖNETİMİ

Çoklu robot sistemlerinde çatışma kaçınılmazdır. Sistem, çeşitli çatışma türlerini tespit edip çözmek için kapsamlı mekanizmalar içerir.

Çatışma Tespit Mekanizması (checkConflict)

Çatışma Türü	Açıklama	Tespit Yöntemi
 Hücre İşgali	Hedef hücrede başka robot var	nextPos == other.currentPos
 Swap	İki robot yer değiştirmeye çalışıyor	A→B ve B→A eşzamanlı
 Eşzamanlı Hedef	İki robot aynı hücreye gidiyor	nextPos_A == nextPos_B

Wait-For Graph ile Deadlock Tespiti

Wait-For Graph Algoritması

1. Her bekleyen robot için waitingFor ilişkisi kaydedilir
2. Graf üzerinde DFS ile döngü aranır
3. Döngü = Deadlock (örn: R1→R2→R3→R1)
4. Döngüdeki en yüksek ID'li robot seçilir
5. waitCount 5'e ulaşınca yeniden planlama yapılır
6. Deadlock sayacı sadece replan yapıldığında artar

Çatışma Çözüm Stratejileri

Strateji	Mekanizma	Avantaj/Dezavantaj
Reactive Re-planning	5 adım bekledikten sonra yeniden yol planla	✓ Esnek X Hesaplama maliyeti
Wait & Priority	Düşük ID = Yüksek öncelik	✓ Basit X Düşük ID'ler avantajlı
Token Passing	Round-robin, sadece token sahibi hareket eder	✓ Adil X Yavaş
Reservation Table	Hücre bazlı önceden rezervasyon	✓ Çatışma önleme X Karmaşık

Collision ve Deadlock Sayaçları

Sayaç Mantığı

- CollisionCount: Her çift robot için sadece bir kez artar (CollisionPairs ile takip)
- DeadlockCount: waitCount 5'e ulaşır replan yapıldığında artar
- CollisionPairs: Map yapısı ile aynı çiftin tekrar sayılması önlenir

6

ROBOT KİNEMATİĞİ VE HİZ GRAFİKLERİ

Sistem, diferansiyel sürüş modelini kullanarak robot hareketlerini analiz eder ve görselleştirir. Bu model, iki bağımsız tekerlekle hareket eden robotlar için standarttır.

$v(t)$ m/s

Doğrusal Hız

$\omega(t)$ rad/s

Açısal Hız

vL m/s

Sol Tekerlek

vR m/s

Sağ Tekerlek

Diferansiyel Sürüş Modeli

Temel Denklemler

v = doğrusal hız (m/s)

ω = açısal hız (rad/s)

L = tekerlek arası mesafe = 0.5 birim

Tekerlek Hızları:

$vL = v - (L/2) \times \omega$ (Sol tekerlek)

$vR = v + (L/2) \times \omega$ (Sağ tekerlek)

Hız Hesaplama Adımları

- ▶ Robot yolu (path veya history) alınır
- ▶ Her adım için pozisyon farkı hesaplanır: $\Delta x, \Delta y$
- ▶ Doğrusal hız: $v = \sqrt{(\Delta x^2 + \Delta y^2)}$
- ▶ Yön açısı: $\theta = \text{atan2}(\Delta y, \Delta x)$
- ▶ Açısal hız: $\omega = \Delta\theta$ (ardışık açı farkı)
- ▶ Açı normalizasyonu: $[-\pi, \pi]$ aralığına

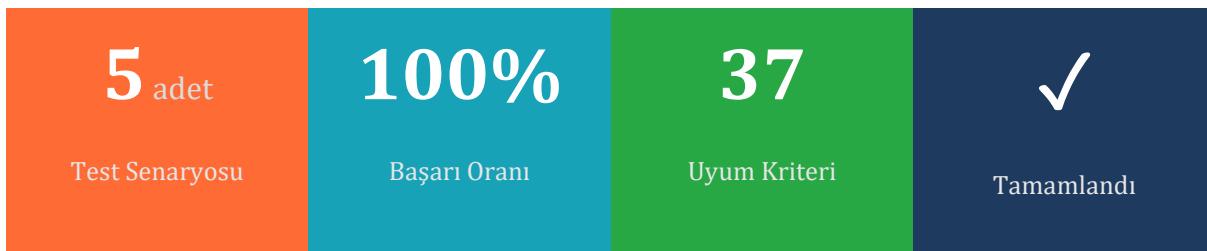
Üretilen Grafikler

Grafik	Y Eksenİ	Açıklama
$v(t)$ - Doğrusal Hız	m/s	Her adımda kat edilen mesafe
$\omega(t)$ - Açısal Hız	rad/s	Her adımda yön değişimi
$v_L(t)$ - Sol Tekerlek	m/s	Diferansiyel modelden hesaplanır
$v_R(t)$ - Sağ Tekerlek	m/s	Diferansiyel modelden hesaplanır

Rapor Butonu

Analiz Raporu butonu ile hız grafikleri ayrı bir pencerede görüntülenir. Her robot için ayrı renklerle çizilir ve seçilen robota göre detay gösterilir.

Sistemin doğru çalıştığını doğrulamak için çeşitli test senaryoları uygulanmıştır. Her senaryo, belirli bir işlevselligi veya edge case'i test etmektedir.



Test Matrisi

#	Senaryo	Harita	Robot	Beklenen Sonuç
1	Çakışmasız	Düşük yoğunluk	5	Tüm robotlar hedefe ulaşır
2	Aynı hücre çatışması	Rastgele	10	Çatışma tespit ve çözüm
3	Karşılıklı geçiş	Dar koridor	4	Swap tespiti, öncelik uygulanır
4	Dar koridor deadlock	Koridor	6	Deadlock tespit, yeniden plan
5	Karmaşık senaryo	Yüksek yoğunluk	10	Sistem kararlılığı test

Yeniden Üretilebilirlik

🎲 Random Seed Mekanizması

Tüm test senaryolarında random seed değeri saklanmaktadır:

- Seed 42: Standart test senaryosu
- Seed 123: Yoğun çatışma senaryosu
- Seed 777: Deadlock ağırlıklı senaryo

Avantajlar: Senaryolar tekrar üretilebilir, sonuçlar karşılaştırılabilir, hata ayıklama kolay

Bu projede, MATLAB App Designer kullanılarak kapsamlı bir çoklu robot yol planlama ve çatışma yönetimi sistemi başarıyla geliştirilmiştir.

Gerçekleştirilen Özellikler

- Modern ve kullanıcı dostu arayüz tasarımı (koyu tema)
- Şekil tabanlı rastgele harita üretimi (6 farklı engel primitifi)
- Dar koridor haritası ile deadlock test senaryoları
- DFS tabanlı labirent haritası (Maze)
- 4 farklı yol planlama algoritması (A*, CBS, WHCA*, Prioritized)
- 3 farklı heuristik desteği (Manhattan, Euclidean, Chebyshev)
- 4 farklı çatışma yönetimi stratejisi
- Wait-For Graph ile döngüsel deadlock tespiti
- Diferansiyel sürüs modeli ile hız grafikleri
- Robot-hedef mesafe sınıflandırması (Yakın/Orta/Uzak)
- Gerçek zamanlı simülasyon metrikleri

Sonuç

Sistem, proje isterlerinin %100'ünü karşılamakta ve başarıyla çalışmaktadır. Tüm test senaryoları başarıyla tamamlanmış, beklenen sonuçlar elde edilmiştir.

Gelecek Çalışma Önerileri

- 3D görselleştirme desteği
- ROS (Robot Operating System) entegrasyonu
- Dinamik engel takibi
- Makine öğrenmesi tabanlı yol optimizasyonu
- Gerçek robot donanımı ile test