ROBOT PROJESİ

Yaşar Can Çilingir

Kocaeli Üniversitesi Bilgisayar Mühendisliği

yasarcanccd@gmail.com

*Özetçe*—Robot projesi kullanıcının başlangıç konumunu belirterek oluşturduğu robot nesnesine (hibrit, paralel, seri, paletli, tekerlekli, spider) robotun özelliklerine göre komut vermesi ile çalışır (manipülatör robot ise yükü sabit kaldığı yerden taşıma, gezgin robot ise hareketli olarak taşıma, hibrit ise kullanıcı tercihine göre taşıma). Komutlar ileri, geri, sag ve sol olarak kullanıcıdan alınır. Her komutun icrası oluşturulan ızgara üzerinde gösterilir. Manipülatör robotlar kol uzunluğundan uzağa ve ızgara dışına yük taşıyamaz. Gezgin robotlar ızgara dışına yük taşıyamaz ayrıca spider robotlar engellerden geçemez. Tekerlekli ve paletli robotların engel geçme süreleri vardır. Hibrit robotlar sabit komutu girilene kadar gezgin robot davranışı gösterir, sabit komutu girildikten sonra manipülatör robot davranışı gösterir.

Anahtar Kelimeler — Kalıtım; Hibrit; Robot;

**1. GİRİŞ**

Robot projesi, belli başlı yöntemlerin (Encapsulation, Inheritance, Polymorphism, Abstraction) gerekli olanları kullanılarak aralarında hiyerarşik bir ilişki oluşturulan robot sınıflarının kullanıcıdan aldığı komutlara göre icralarını grafik yardımıyla kullanıcıya gösterilmesini sağlar.

Robotun başlangıç konumu kullanıcıdan alınır ve ızgarada konuma denk gelen kısıma yerleştirilir. Ardından robotun tipine göre ileri, geri, sag, sol ve sabit işlevleri gerçekleştirilir.

**2. TEMEL BİLGİLER**

Proje JAVA programlama dili kullanılarak yazılmıştır

Proje gelişiminde tümleşik geliştirme ortamı olan Netbeans IDE kullanılmıştır.

Grafik ile oluşturulan ızgaranın sol üst köşesi (1,1) konumu olarak kabul edilmiştir. X ekseni saga doğru gidildikçe atmaktadır, Y ekseni ileri doğru gidildikçe artmaktadır. Her bir ızgara karesi 10 m² olarak Kabul edilmiştir.

Kullanıcıdan alınan hızlar m/s cinsinden kabul edilmektedir.

JAVA programlama dili çoklu kalıtımı (multiple inheritance) doğrudan desteklemediği için interface ve abstract class yapıları kullanılmıştır.

**3. TASARIM**

Proje aşağıdaki alt başlıklar altında geliştirilmiştir.

***3.1 Hiyerarşik yapı***

Robot projesinde oluşturulabilir, düzenlenilebilir 6 tane robot sınıfı(tekerlekli, paletli, spider, hibrit, paralel, seri) bulunmaktadır. Bu 6 robot sınıfının hiyerarşik olarak özelliklerini aldığı 2 üst sınıf (gezgin, manipulator) vardır. Tekerlekli, paletli ve spider robotlar sadece gezgin üst sınıfına bağlıdır. Paralel ve seri robotlar sadece manipulator üst sınıfına(interface) bağlıdır. Hibrit robotlar hem gezgin üst sınıfından hem de manipulator üst sınıfından özelliklerini alır. Tüm sınıfların atası olan Robot sınıfı(interface) ise gezgin ve manipulator üst sınıflarına özelliklerini aktarır, dolayısıyla Robot sınıfı, kullanılabilir tüm robot sınıflarına (tekerlekli, spider, paletli, parallel seri, hibrit) özelliklerini aktarmış olur. (Resim 7.1)

***3.2 Yöntemler***

Projede kullanılan methodlar aşağıda açıklanmıştır.

robotTipiAl() : Izgara üzerinde gösterilecek olan robotun tipi belirlenir.

engelSayisiAl() : Izgara üzerine kaç engel yerleştirileceğinin sayısı alınır.

engelKoordinat() : Izgara üzerine yerleştirilen engellerin koordinatları alınır.

gezginStartKoordinat() : Izgara üzerine yerleştirilecek robotun başlangıç koordinatları alınır.

gezginSonKoordinat(): Gezgin robotun bulunduğu son koordinatı hesaplar.

hibritSabitleme() : Sabit işlemi girildikten sonra hibrit robot manipulator robot işlevi görür.

yukSonKoordinat() : Manipulator robotun taşıdığı yükün son koordinatını hesaplar.

tasimaMesafesiKontrol() : Manipülatör robotun kol uzunluğu aşılıp aşılmadığını control eder.

sureGoster(): Gezgin robotun toplam hareket süresini gösterir.

sure2Goster(): Manipülatör robotun taşıma hızına göre ve süratine göre toplam süreyi gösterir.

sure3Goster(): Hibrit robotun hareket, taşıma ve sabitlenme süresinin toplamını gösterir.

ciktimi(): Robotların ızgaradan çıkıp çıkmadığını kontrol eder.

engelKontrol2(): Spider robotun engele değip değmediğini kontrol eder.

engelKontrol(): Gezgin robotların engele değip değmediğini kontrol eder.

paint(): Izgaranın çizilmesini sağlar.

***3.3 Sınıfların Özellikleri***

Robot(Interface):

Tüm sınıfların atasıdır.

Motor sayısı, yuk miktarı ve robot tipi için bodysiz methodlar yazılmıştır.

Gezgin Robot(abstract class):

Tekerlekli, paletli, spider, hibrit robotlarının atasıdır.

Tekerlekli, paletli, hibrit ve spider robotlarının ortak değişkenleri burda tanımlanmıştır ayrıca tekerlekli, hibrit ve paletli robot için engel geçme süresi methodu burada tanımlanmıştır.

Manipulator Robot(Interface):

Seri, hibrit ve paralel robotlarının atasıdır.

Seri, hibrit ve paralel robotlar içinde ortak kullanılacak methodlar burada tanımlanmıştır.

Tekerlekli Robot(class):

Gezgin Robotlardan miras aldığı değişkenleri kullanır.

Engel geçme süresi (motor sayısı \* 0.5) dir.

Paralel Robot(class):

Gezgin Robotlardan miras aldığı değişkenleri kullanır.

Engel geçme süresi (motor sayısı \* 3 ) dür.

Spider Robot(class)

Gezgin Robotlardan miras aldığı değişkenleri kullanır.

Engellerden geçemez.

Paralel Robot(class):

Manipülatör robotlardan miras aldığı methodları kullanır.

Yük kapasitesi ve hızı seri robotlardan hızlıdır.

Seri Robot(class):

Manipülatör robotlardan miras aldığı methodları kullanır.

Yük kapasitesi ve hızı paralel robotlardan düşüktür.

**4. KARŞILAŞILAN SORUNLAR VE ÇÖZÜM YÖNTEMLERİ**

**1.** JAVA programlama dili multiple inheritance yapısını doğrudan desteklemediği için çoklu kalıtım için interface yapısı araştırıldı, Interface yapısı ile değişkenleri doğrudan aktarma sağlanamadı. Interface yapısı ile kullanılacak ortak methodlar tanımlanarak çoklu kalıtım sağlandı.

**2.**  Izgara sınırlarını aşma durumu kontrolü için çizdirilen ızgaranın sınırları belirlendi, robotun ızgaradan çıkma durumunda sınırda kalması için çıktığı ızgara sınırlarının koordinatları kullanıldı

**3.** Manipülatör robot kol uzunluğu aşıldığında süre hesabı için aşılan mesafede ekleniyordu. Bu durumun çözümü için kol uzunluğu aşıldığında süre hesabı için kol uzunluğu kullanıldı ve grafik olarak yükün kol uzunluğundan daha uzakta gösterilmesi engellendi.

**4.** Hibrit robotun işlemleri için sıfırdan kodlama yapılmak yerine sabitlenene kadar gezgin robot methodları, sabitlendikten sonra manipülatör robot methodları kullanıldı.

**5.** Spider robotun engel ile karşılaştıktan sonra programın bitmesi ve robotun konumunun engel üzerinde gösterilmesi için karşılaştığı engelin koordinatları kullanıldı.

**6.** Kullanıcının girdiği her adımın ızgara üzerinde gösterilmesi için her komut girildiğinde konum ve çizimi yeniden güncellendi.

**5. KAZANIMLAR**

Robot projesinin bana kazandırdıkları:

1.Inheritance kullanımı

2. Abstract class yapısı, interface ile ilişkisi

3. JAVA programlama dili ile çoklu kalıtım yapmak

4. Classlar arası işlem yapmak

5. JAVA programlama dilinde grafik kütüphanesinin kullanımı

**6.PSEUDO KOD**

1-Başla

2-Oluşturulacak robot sayısını al

3-Oluşturulan robotun tipini al

4-Motor sayısını al

5-Yük miktarını al

6-if(tekerlekli==robot tipi)

7-Teker sayısını al

8-Gezinme hızını al

9-while(gezinme hızı< tekerlekli min hız)

10- Gezinme hızını al

11-if(paletli==robot tipi)

12-Palet sayısını al

13-Gezinme hızını al

14-while(gezinme hızı< Paletli min hız)

15- Gezinme hızını al

16-if(spider==robot tipi)

17-Bacak sayısını al

18-Gezinme hızını al

19-while(gezinme hızı< Spider min hız)

20- Gezinme hızını al

21-if(paralel==robot tipi)

22-Yuk miktarını al

23- while(yuk miktari > yük kapasitesi)

24- Yük miktarını al

25-Kol uzunlugunu al

26- Taşıma hızını al

21-if(seri==robot tipi)

22-Yuk miktarını al

23- while(yuk miktari > yük kapasitesi | | paralel yuk miktarı< yuk miktarı)

24- Yük miktarını al

25-Kol uzunlugunu al

26- Taşıma hızını al

27-if(hibrit==robot tipi)

28- Hibritin gezgin ve manipulator kısmını al

29- Gezinme hızı, taşıma hızı, kol uzunlugu al

30- Kaçıncı robotun problemi çözeceğini al

31- Engel sayısı al

32- Engel koordinatlarını al

33- Robotun başlangıç koordinatlarını al

34- Paint methodu ile engelleri ve başlangıç konumunu çiz

35- while(komut!=cikis)

36- Kullanıcıdan yön al

37- Yönü başlangıç koordinatlarına ekle

38-if(robot tipi == seri | | robot tipi == paralel | | hibrit.manipulatör)

39- kol uzunlugu kontrol et

40- if(robot tipi== spider | | robot tipi== tekerlekli | | robot tipi== paletli | | hibrit.gezgin)

41- engel koordinatını kontrol et

42- if(robot tipi == spider & & başlangıç koordinatı == engel)

43- spider robot engeli geçemez, süreyi yazdır,break;

44-if(robot tipi == seri | | robot tipi == paralel)

45-if(başlangıç koordinat + yön > kol uzunlugu | | başlangıç koordinat + yön > ızgara koordinatları)

46- süreleri yazdır, break

47- if(başlangıç koordinat + yön > ızgara koordinatları)

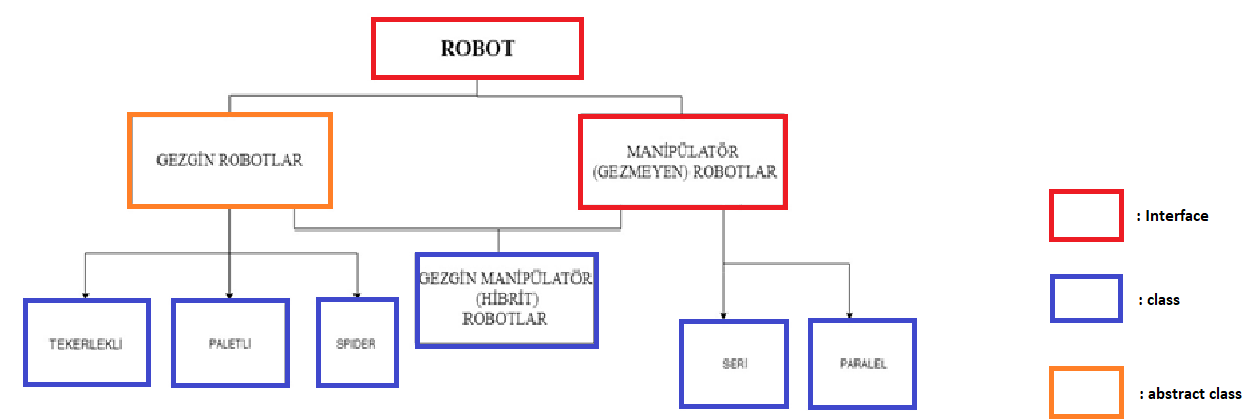
48- süreleri yazdır, break;

44- if(komut == cikis)

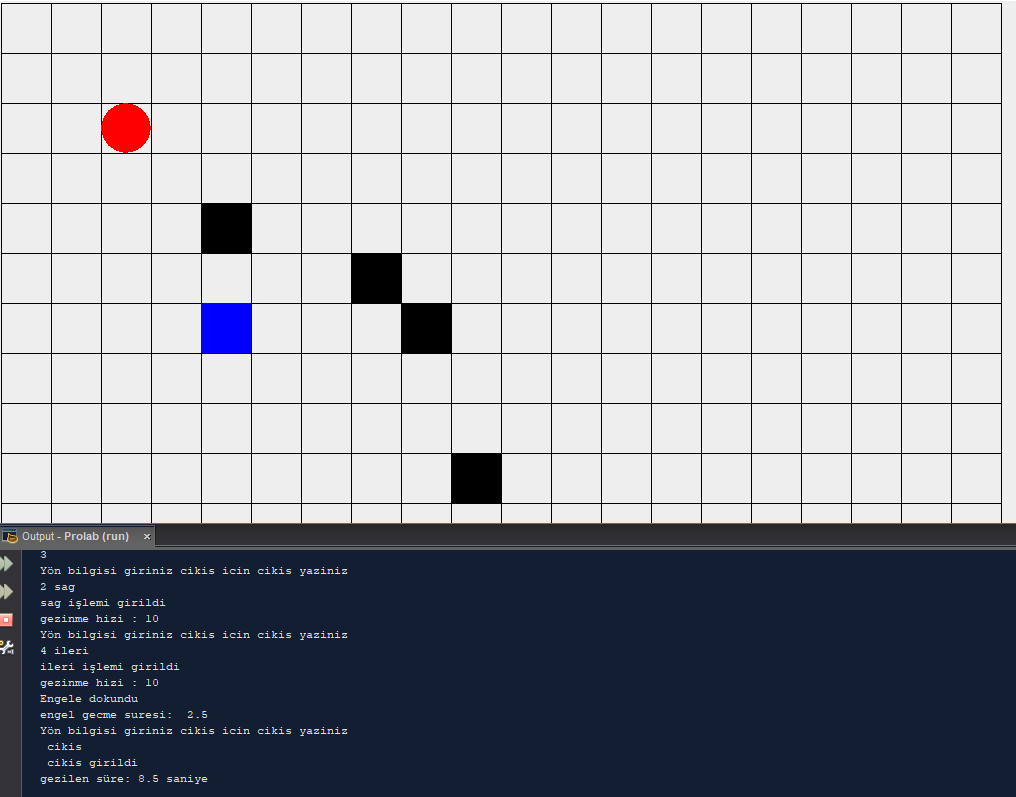
45- süreleri yazdır.

**7. RESİMLER VE EKRAN GÖRÜNTÜLERİ**

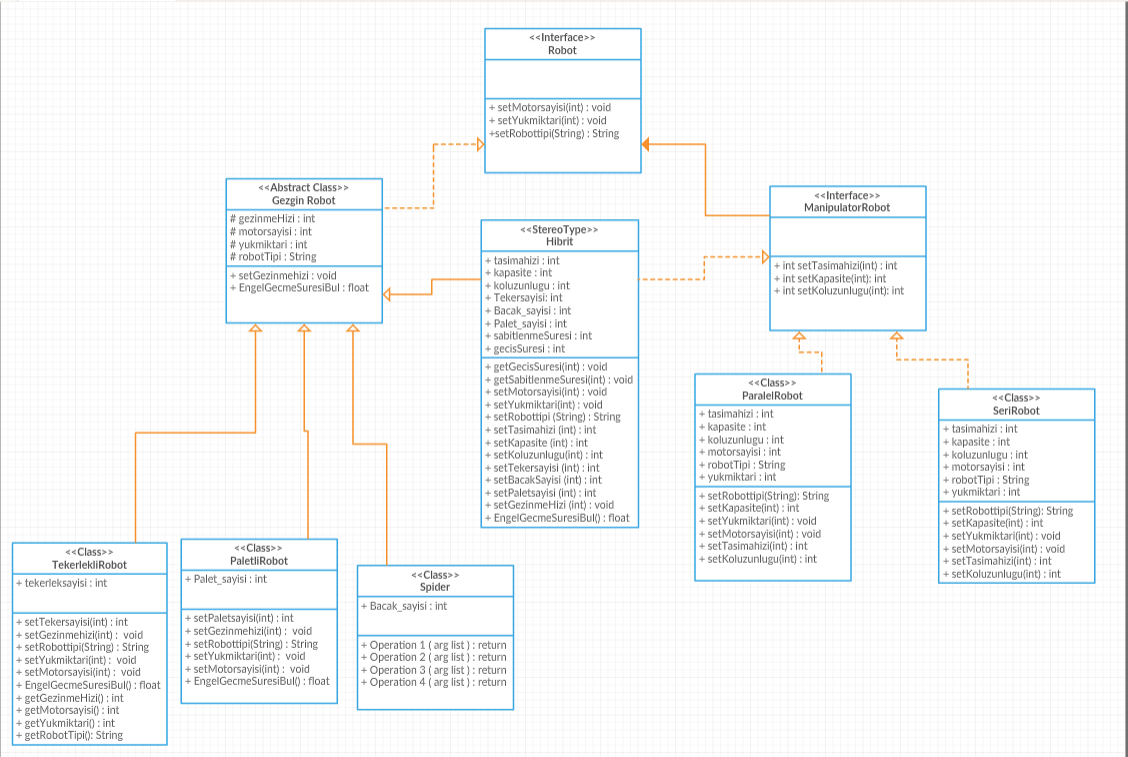
**7.1**

****

**7.2**

****

**8. UML DİAGRAMI**

****

**9. KAYNAKÇA**

JAVA swing, Jframe kullanımı

<https://www.youtube.com/watch?v=NaPSFjb8YAM>

(Erişim tarihi : 05.12.2018)