ProLab2 Gezgin Robot Projesi

1st GÜRKAN TÖNGEL
Kocaeli Üniversitesi
Bilgisayar Mühendisliği
210202032

I. Açıklama

Kocaeli Üniversitesi Bilgisayar Mühendisliği Programlama Laboratuvarı2 dersi 1. proje Gezgin Robotlar Projesi hakkında rapor yazısı

II. Özet

A. Proje Açıklaması

Proje belirli kurallara göre hareket eden bir robotun önündeki engelleri aşarak istenen hedefe ulaşmasını sağlayan bir oyun projesidir. Oyunda bulunan iki adet problemin çözülmesi beklenmektedir. Problemlerin çözümü için nesneye yönelik programlama ve veri yapıları bilgilerinin kullanılması beklenmektedir

B. Geliştirme Süreci

Geliştirme süreci içerisinde öncelikle projenin nesne yönelimli programlamaya ve veri yapılarına uygun olmasına dikkat edilmiştir. Projenin menüsü ve sonuç ekranı Python pyqt kütüphanesi ile, labirent kısmı pygame kütüphanesi ile geliştirilmiştir. Random labirent oluşturma prim algoritması kullanılarak geliştirilmiştir.

III. GİRİŞ

Projede 2 adet problemin çözülmesi istenmektedir. Problem 1 de bize önceden verilen url deki matrise göre engeller olan labirent oluşturup grafiksel bir ortamda çözmemiz istenmektedir. Problem2 de ise kullanıcıdan alınan satır ve sütun değerlerine göre random bir labirent oluşturup labirenti çözmemiz istenmektedir. Problem 1 de program her çalıştırma da verilen linkten güncel matris çekilir ve labirent oluşturulur. Matrislerde 1*1, 2*2 ve 3*3 olmak üzere 3 farklı engel bulunmaktadır. Engel tiplerindeki farklılığı göstermek için her engel farklı bir resim ile belirtilmiştir. Değiştir butonuyla ise url ler arasında geçiş yapılır ve diğer matris ile labirent tekrrar oluşturulur. Labirent 2 de ise Prim algoritması kullanılarak 1*1 lik engeller ile rastgele bir labirent oluşturulur. Değiştir butonuyla ise tekrar random bir labirent oluşturulur.

IV. Yöntem

A. Geliştirilme Ortamı

Projenin geliştirilmesinde Python dili kullanılmış, grafiksel arayüz pyqt ve pygame kütüphaneleri ile geliştirimiş, IDE olarak VS Code tercih edilmiştir.

B. Problem1 urlden matris oluşturulması

Bu projede bize 2 adet url verilip program her çalıştırıldığında bu url deki bilgilere göre labirent oluşturmamız istenmiştir. Program her çalıştırıldığında url deki matrisler kaydedilip arayüzdeki değiştir butonuna tıklandığında url ler arası geçiş sağlanacak şekilde proje geliştirilmiştir. Url lerden matris bilgilerinin alınması python urllib kütüphanesi ile sağlanmıştır

C. Problem1 Engel Tipleri

Bize verilen url lerde 1*1, 2*2 ve 3*3 olmak üzere 3 farklı engel tipi olacağı söylenmiş, bu engel boyutlarına sığacak şekilde istediğimiz gibi engel tasarlayabileceğimiz belirtilmiştir. Bu engel tipleri arasındaki fark anlaşılması için her engel tipi arayüzde farklı bir image ile gösterilmiştir.

D. Problem2 Random Labirent

Bu projedeki problem 2 bölümünde ise kullanıcıdan alacağımız satır ve sütun boyutlarında rastgele labirent oluşturup çözmemiz istenmiştir. Çalıştır butonu ile labirentteki karelerin bulutlanması ve labirentin tamamen görünmeden sadece robotun çevresindeki karelerin görünerek labirentin çözülmesi istenmektedir. Random labirent oluşturulması Prim Algoritması ile geliştirilmiş, çözülmesi ise öncelikli yönler ile kareler gezilerek çıkış bulunmaya çalışılacak şekilde geliştirilmiştir. Robot başlangıçta çıkış konumunu bilmemektedir.

E. Prim Algoritması ile Random Labirent Oluşturma

Metod, ”maze” adında bir labirent nesnesi alır. İlk olarak, labirentin tüm hücrelerini 1 ile sıfırlar ve ”width” ve ”height” adında iki değişkene labirentin genişliği ve yüksekliğinin yarısını atar. Daha sonra, labirentin içindeki bir rastgele hücreyi ”startx” ve ”starty” adlı değişkenlere atar ve bu hücreyi ziyaret edilmiş olarak işaretler.Döngü, ”checklist” adlı bir liste içindeki hücrelerin tamamı ziyaret edilene kadar devam eder. Her döngüde, ”checklist” içindeki rastgele bir hücre seçilir ve ”checkneighbors” adlı bir yardımcı fonksiyon ile kontrol edilir. Bu fonksiyon, seçilen hücrenin komşuları arasında ziyaret edilmemiş hücrelerin var olup olmadığını kontrol eder ve varsa bu hücreleri ”checklist” listesine ekler. Eğer seçilen hücrenin komşuları arasında ziyaret edilmemiş hücre yoksa, o hücre ”checklist” listesinden çıkarılır. Bu işlem, labirentte tüm

hücreler ziyaret edilene kadar devam eder ve sonuçta labirent, Prim algoritması ile rastgele şekilde oluşturulur.

V. Deneysel Sonuçlar

A. Program Menüsü

Program çalıştırıldığı zaman bizi menü karşılar.

B. Problem2 için satır ve sütun girdisi

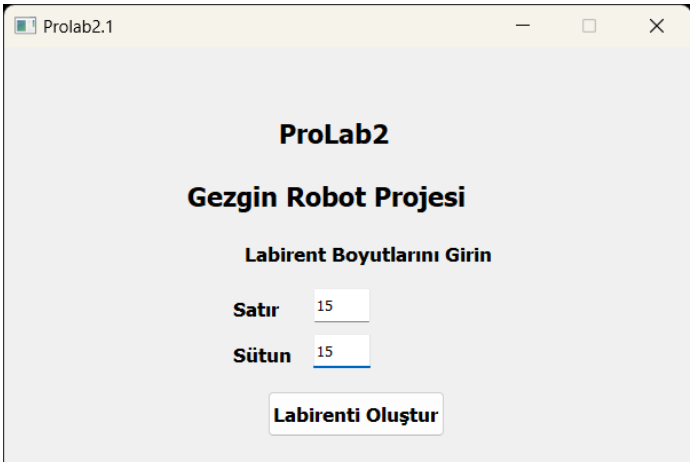


Fig. 1. Random Labirent için satır sütun girdisi alınır

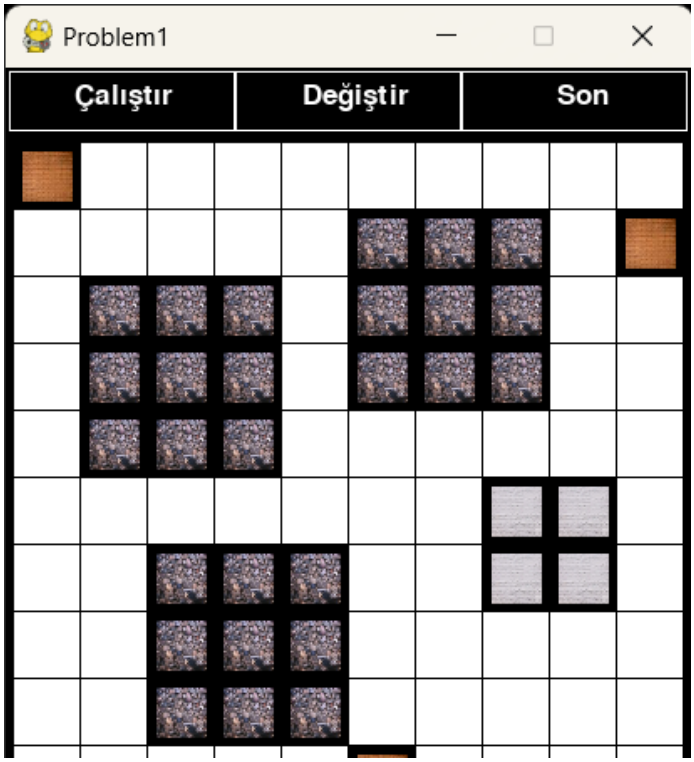
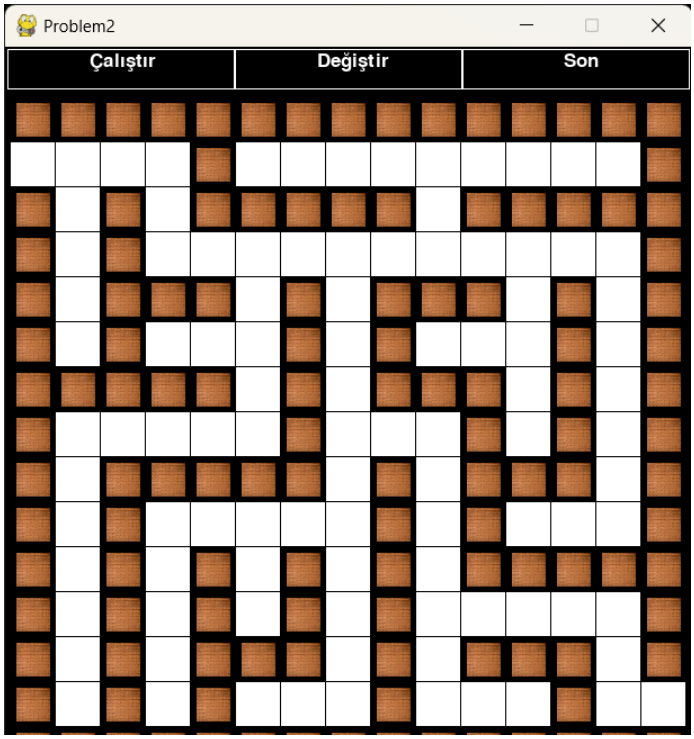


Fig. 2. Örnek url1 labirent

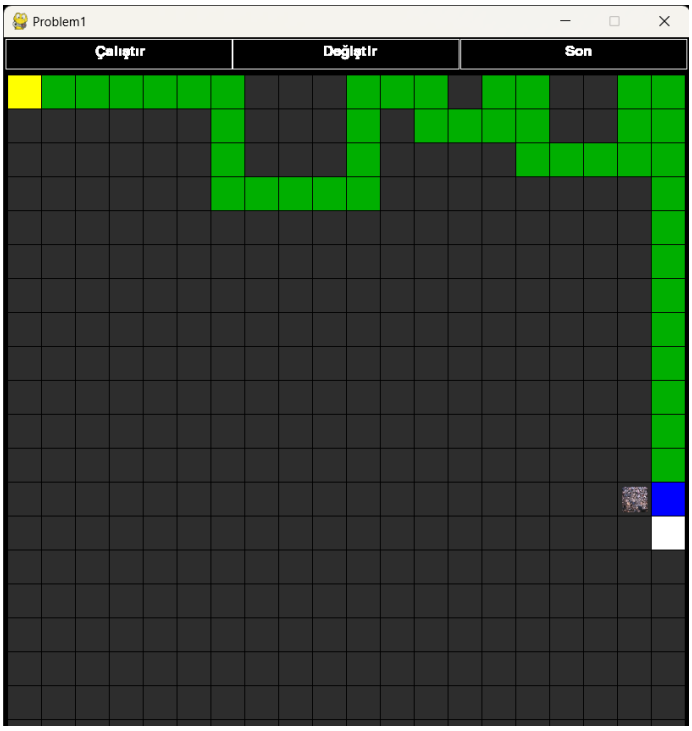
C. Problem1 örnek url1

D. Problem1 örnek url2



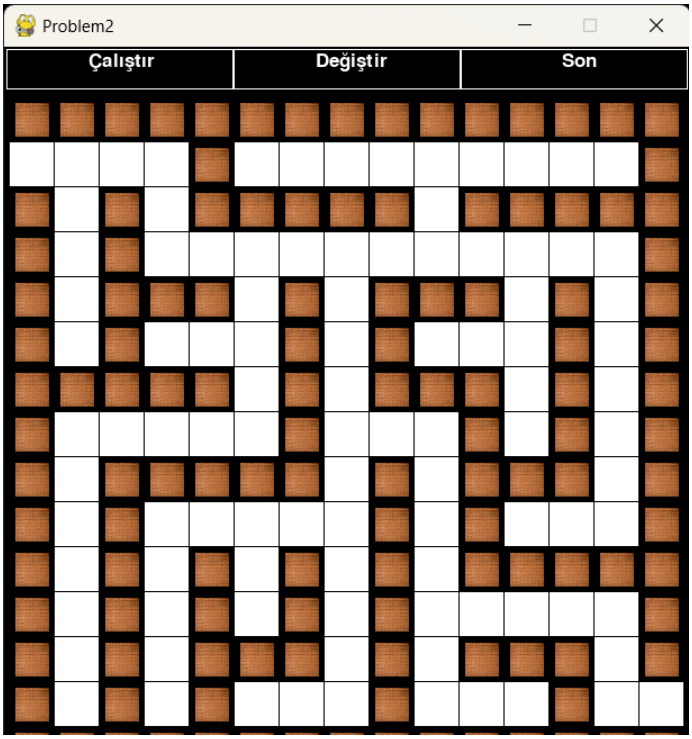
Örnek url2 labirent

E. Labirent çözüm anı



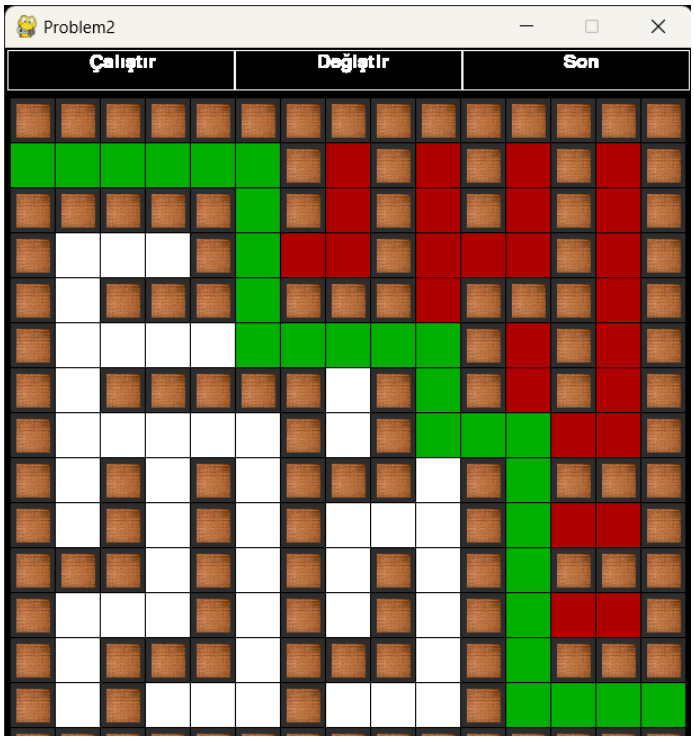
Gezilmeyen kareler bulutlu bir şekilde robot gezdiği karelere iz bırakır

F. Problem2



Problem2 random labirent

G. Labirent Çözüm Anı

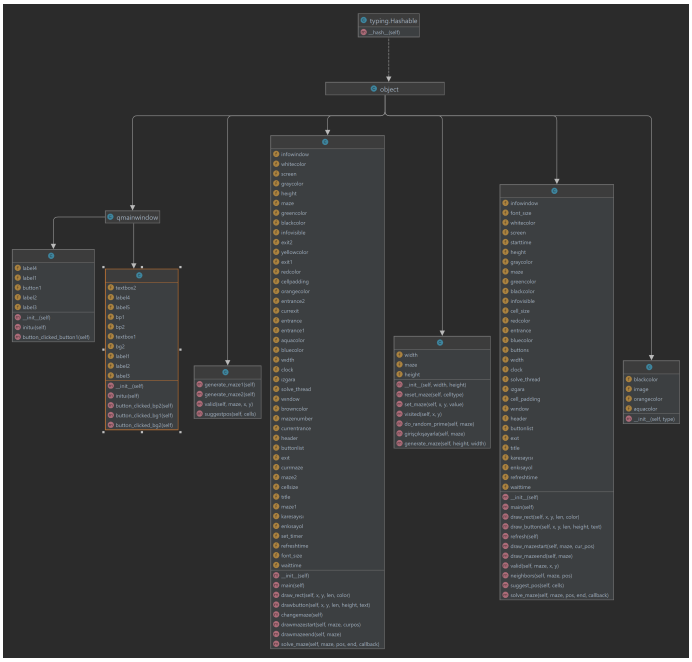


Random labirent çözümü

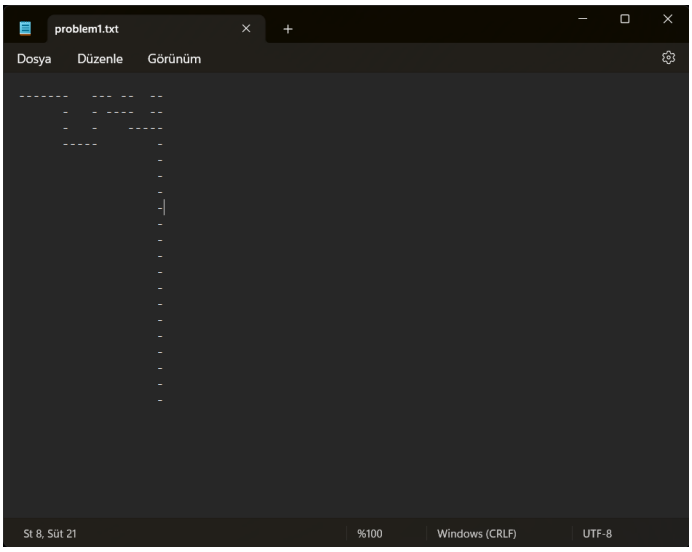
H. Gezilen Kare ve Süre Bilgileri

Gezilen kare, en kısa yol ve gezme süresi gösterimi

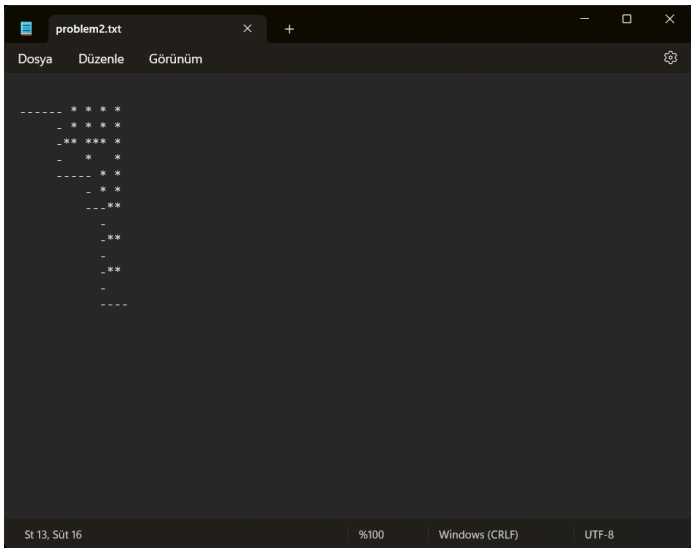
I. UML Diyagramı



J. Problem1 gezilen yolların text dosyasına kaydedilmesi



K. Problem2 gezilen yolların text dosyasına kaydedilmesi



L. Proje UML Diyagramı

VI. Katkılar

Öncelikle bir uygulamayı geliştirirken öncelikle zaman planlaması ve bir uygulamayı geliştirirken hangi parçaların, bölümlerin önemli öncelikli olduğunu fark ettik. Böylelikle uygulama geliştirme sürecinde geliştirmenin nasıl ve ne sırayla yapılması gerektiğini öğrendik. Ayrıca programlı ve düzenli çalışmanın geliştirme sürecini çok daha kolaylaştırdığını öğrendik. İşe başlamadan önce araştırma yapmanın gerekliliği, hiç bir bilgi birikimi olmadan uygulama geliştirmenin imkansızlığını ve bilinçli ve bilgi birikimiyle uygulama geliştirmeye girmenin gerekliliğini kavradık.

Bu uygulamayı geliştirirken nesne yönelimli programlama ve veri yapıları mantığını kavradımk ve grafik arayüzü tasarlamada tecrübeye edindik ve kendimizi geliştirdik.Hata ayıklama sürecinin çok sancılı, uzun bir süreç olduğunu fark ettik. BT alanında gelişimin ve paylaşımın neredeyse her şey olduğunu anladık.

VII. Sonuç

Sonuç olarak veri yapıları algoritmaları ile random labirent oluşturma, grafiksel bir oyun yapma konusunda tecrübe edindim. Geliştirme sürecinde nesne yönelimli programlama ve veri yapıları mantığını ve bir kullanıcı arayüzünün nasıl geliştirildiğini kavramış oldum.

References

[1] https://www.youtube.com/watch?v=jO6qQDNa2UY
[2] https://www.youtube.com/watch?v=AY9MnQ4x3zk
[3] https://www.youtube.com/watch?v=i6xMBig-pP4&list=PLzMcBGfZo4-lp3jAExUCewBfMx3UZFkh5
[4] https://www.youtube.com/watch?v=qYomF9p_SYMx
[5] https://www.pygame.org/wiki/tutorials
[6] https://www.youtube.com/playlist?list=PLqG356ExoxZUGztzAxqIWkkTq8JVa-o3X
[7] https://www.geeksforgeeks.org/pygame-tutorial/
[8] https://www.javatpoint.com/pygame
[9] https://ryanstutorials.net/pygame-tutorial/
[10] https://coderslegacy.com/python/python-pygame-tutorial/

[11] https://www.tutorialspoint.com/pygame/index.htm
[12] https://www.tutorialspoint.com/pyqt/index.htm
[13] https://wiki.python.org/moin/PyQt/Tutorials
[14] https://www.pythonguis.com/pyqt5-tutorial/
[15] https://realpython.com/python-pyqt-gui-calculator/l
[16] https://www.pythontutorial.net/pyqt/
[17] https://build-system.fman.io/pyqt5-tutorial
[18] https://www.guru99.com/pyqt-tutorial.html
[19] https://pythonprogramming.net/basic-gui-pyqt-tutorial/
[20] https://bilgisayarkavramlari.com/2007/12/24/dijkstra-algoritmasi/

a) SÖZDE KOD: –Problem Seçme Menüsü

– Problem1 seçilmiş ise:
1- Url1 text dosyası indirilip matris, giriş ve çıkış konumları belirlenir
2- Url2 text dosyası indirilip matris, giriş ve çıkış konumları belirlenir
3- Varsayılan matris url1 olmak üzere program çalıştırılır
4- 3 farklı engel tipi farklı olarak arayüzde gösterilir
5- Çalıştır butonuna basıldığında kareler bulutlanır ve çözüm başlar
6- Değiştir butonuna basıldığında urller arası geçiş yapılır
7- Kare dolaşma önceliği sağ, aşağı, sol, yukarı olacak şekilde labirent çözülmeye çalışılır
8- Eğer çıkış noktasına ulaşılmışsa oyun biter ve labirentin çözülme süresi, gezilen kare sayısı ve çözülebilecek en kısa yol ekranda gösterilir.

–Problem2 seçilmiş ise:
1- Kullanıcıdan satır ve sütun değerleri input olarak alınır
2- Alınan satır ve sütun boyutlarında random labirent oluşturulur
3- Labirentteki engeller problem1 deki 1*1 lik engel tipinde oluşturulur.
4- Değiştir butonuna basılırsa tekrar random bir labirent oluşturulur
5- Çalıştır butonuna basılırsa kareler bulutlanır ve sadece robotun çevresi gözükecek şekilde labirent çözülmeye başlar
6- Robotun labirent gezmesi simüle edilebilmesi için her kare arasında belirlenen sürede bir bekleme uygulanır. 7- Son butonuna basılırsa bekleme süresi sıfır olur ve robot labirentin sonuna gider 8- Labirent çözüm süresi, gezilen kare sayısı ve gidilebilecek en kısa yol bilgileri ekranda gösterilir