ŞİRİNLER

Niyazi Şahin, Eren Cem AY
190202069 - 190202063
Kocaeli Üniversitesi Mühendislik Fakültesi
Bilgisayar Mühenedisliği(İÖ)
niyazisahin3800@gmail.com, erencemayez@gmail.com

1. Problem Tanıtımı

Öncelikle uygulama bir şirinler oyunudur. Kullanıcı kontrollü olarak hareket özelliği bulunan şirin, düşman karakterler olan azma ve gargamelden kaçarak "Şirine" karakterinin bulunduğu noktaya gitmeyi amaçlamaktadır. Kullanıcı seçtiği şirin ile şirineye ulaşmaya çalışırken harita üzerinde çıkan ekstra puan amaçlı objeleri toplama özelliğine sahiptir. Oyunda düşman karakterlerin hareketi program tarafından Dijikstra Algoritması kullanılarak en kısa yol otomatik olarak hesaplanmış ve görsel olarak sunulmuştur.

2. Giriş

Yukarıda belirtildiği gibi proje bir şirinler oyunu projesidir. Oyunda amaç seçilen şirin karakterinin en fazla puan toplayarak ve düşman karakterlere yakalanmadan şirineye ulaşmaktır. Programın yazılımında ki amaç ise düşman karakterlerin seçilen şirin karakterine gideceği en kısa mesafeli yolu Dijikstra Algoritması ile hesaplamaktır.

Kullanıcı oyun girişinde kendisine sunulan iki karakterden, yani tembel şirin ve gözlüklü şirinden birini seçmek zorundadır. Tembel şirin her tıklamada 1 adım atabilirken gözlüklü şirin her tıklamada 2 adım atabilmektedir. Her iki şirinin de çapraz adım atma özelliği yoktur.

Kullanıcı oyunu oynayacağı şirini seçtikten sonra oyun haritası ile karşılaşacaktır. Kullanıcı haritanın ortasından oyuna başlayacaktır. Seçilmiş bazı köşelerde düşmanlar vardır ve kullanıcı şirini her hareket ettirdiğinde düşmanlar bulundukları yerlerden en kısa yol algoritmasına göre harekete geçeceklerdir. Kullanıcı oyun esnasında belirli süre zarfında haritanın rastgele noktalarında oluşacak mantar ve altınları toplayıp ekstra puan kazanabilir. Her bir mantar 50 puan kazandırırken, her bir altın 5 puan kazandıracaklardır. Ancak düşman karakterler şirine her dokunduklarında kullanıcı 15 puan kaybedecektir. Puan 0 veya altında olduğunda kullanıcı oyunu kaybedecektir.

Kullanıcının hareketine göre adım atacak olan düşman karakterler ile kullanıcı şirin arasındaki en kısa mesafeli yol ekranda yeşil renkte gözükmektedir. Düşman karakterler bu yeşil renkli doğrultuda hareket edeceklerdir. Kullanıcının her hareketi yeşil renkli doğrultuyu değiştirecektir. Çünkü yeşil renkli doğrultu, en kısa mesafenin yolu değiştiği için değişecektir. Yapı dinamik bir yapıdır.

3. Yöntem

Uygulama C++ dilinde yazılmıştır. Grafik kütüphanesi olarak SFML kütüphanesi seçilmiştir.

Program arayüz olarak oluşturacağı haritayı algılayabilmek için bir txt uzantılı dosyaya ihtiyaç duymaktadır. Bu dosyanın adı program içerisinde "harita.txt" olarak adlandırılmıştır. Belirtilen dosya haritayı 1 ve 0 lardan oluşturmaktadır. Harita oluşturma algoritması, hedef dosyadan veriler okunurken 1 ve 0 ların kontrolü şeklinde yapılmıştır. Ayrıca txt dosyasının ilk satırlarında düşman karakterlerin tipi ve başlangıç kapıları yer almaktadır. Program bu bilgileri txt dosyası okunurken karakter karakter okuyup, karakterleri dilimleyerek okumuştur ve program içerisindeki gerekli değişkenlere atamıştır.

Program 1 adet pencere içerisinden yürütülmektedir. Bu pencere SFML kütüphanesinin faydalı fonksiyonları tarafından sağlanmış ve şekillendirilmiştir.

Oyun içerisindeki düşman ve oyuncu karakterler için texture oluşturulmuş ve bu texturelar sprite olarak pencerenin gerekli koordinatlarına yerleştirilmiştir. Oyuncu spriteları klavyedeki yön tuşları aracılığıyla hareket ettirilmektedir. Bu özellik SFML kütüphanesindeki klavye hareket algılayıcı fonksiyonlar tarafından sağlanmıştır. Bu fonksiyonlar çalıştığında, düşman karakterleri hareket ettirmek için oluşturduğumuz bool tipi değişkenler aktif olmakta ve bunun sonucunda düşman karakterler hareket etmektedirler.

Yukarıda belirtildiği gibi oyuncunun her hareketinde klavye hareketi algılanmakta ve bu durum bool tipi kontrol değişkenlerini aktif hale getirmektedir. Bu durum aynı zamanda en kısa yol algoritması olan Dijiksta Algoritması için oluşturulan fonksiyonları da harekete geçirmektedir. Bu sayede en kısa yol algoritması çalışır ve düşman karakterlerin oyuncuya gidebileceği en kısa yol harita üzerinde yeşil saydam renkte gözükür. Bu olayı sağlayan algoritma, programa bağlı liste olarak uygulanmıştır. Bağlı liste sayesinde kontrol edilen her birim bağlı listeye eklenmekte ve en sonunda en kısa olan tespit edilmektedir. Bu işlem dinamiktir ve her harekette kontrol edilmektedir.

Kullanıcı her hareketi esnasında harita üzerinden altın veya mantar toplayabilir. Bunlar rastgele süreli ve rastgele koordinatlı olarak ayarlanmıştır.

4. Sonuç

Program grafikler, oyuncu hareketleri, altın ve mantar hareketleri, oyunun başlangıcı ve bitişi olarak bir bütün halinde gerekli kullanıcı testlerinden geçmiş ve testler başarı ile sonuçlanmıştır.

Programın işleyişinde kullanılan Dijikstra Algoritması aynı şekilde test edilmiş ve olumlu şekilde çalışmıştır.

Oyun grafiksel arayüzü ile kullanıcıya sunulmuştur.

.

A. Sözde kod

```
struct node{
int x,y,dist
node parent(tipi node bağlı olduğu blok)
}

vector bakılmamıs_nodelar ( vectorde distance en
dusuk olan en basta olucak )

for tüm bloklar:
   yürülenebilir ise balkılmamıslara ekle
   x,y = blok pozisyon
   distance sonsuz ( kendi bulunduğu blok 0)
   parent null

while bakılmamıs_nodelar bos degilse:
   suanki_node = en küçük distancelı node
   suanki_nodu vectorden çıkar

   eğer suanki_node son nodesa bitti
         parrentları takip ederek yolu bul
   değilse vectorün içinde olan tüm komşu nodelar
için
         yeni_distance = şuanki_node distancel +
komşu ile distancı

         eğer yeni_distance < suanki_node.dist
               komşunun distancı =yeni_distance
               komşunun parenti = suanki_node
```

5. Referanslar

- [1]https://www.sfml-dev.org/tutorials/2.5/window-window.php
- [2]https://www.sfml-dev.org/tutorials/2.5/window-inputs.php
- [3]https://www.sfml-dev.org/tutorials/2.5/graphics-sprite.php
- [4]https://www.sfml-dev.org/tutorials/2.5/graphics-text.php
- [5]https://en.wikipedia.org/wiki/Dijkstra%27s_algorithm
- [6]https://www.youtube.com/watch?v=eE3qGNdEpEs&t=1363s
- [7]https://www.youtube.com/watch?v=WFts4-62F6U

UML DIAGRAMLARI