Programlama Laboratuvarı 1. Proje

1. Abdulrahman Dülek
Bilgisayar Mühendisliği
Kocaeli Üniversitesi
Kocaeli, Türkiye
dulekabdulrahman@gmail.com

2. Anıl Can Göl
Bilgisayar Mühendisliği
Kocaeli Üniversitesi
Kocaeli, Türkiye
anilcangol@gmail.com

***Abstract*— Bu doküman Kocaeli Üniversitesi Bilgisayar Mühendisliği bölümü Programlama Laboratuvarı dersi 1. proje ödevi için hazırlanmıştır.**

Keywords —word; ieee; maliyet hesaplama; parselleme; kaynak arama.

I. Özet

Bu projede, Curl ve Simple DirectMedia Layer (SDL) gibi kütüphanelerin yardımıyla -kullanıcıdan alınan veriler ışığında- matematiksel hesaplamalar yapılacaktır. Bu hesaplamalar sonucunda, örnek şirket için optimum maliyet ve durumlar hesaplanacak ve çıkarılan sonuçlar kullanıcıya grafik ekranında gösterilecektir.

II. Giriş

A. *Projede amaçlananlar şunlardır:*

Temel amaç, denizlerde doğal kaynak arama ve çıkarma operasyonları yapan şirketimizin ihtiyaçlarını karşılamaktır. Bu sayede şirketin maksimum kâr elde etmesine yardımcı olmaktır. Bahsedilen hedeflere ulaşmak için denizdeki arama bölgesini en etkili ve optimal parçalara bölmek gerekmektedir.

Yazılım, proje dosyasında verilen kurallara bağlı kalarak kâr ve maliyet hesaplamaları yapar. Hesaplamalar sonucunda deniz arama alanını uygun parçalara böler. Bölmeler, her biri farklı boyutlar ve/veya yerlerde olabileceği bir görsel harita üzerinde temsilen gösterilir. Bu harita sayesinde şirket, kaynaklarını en iyi şekilde yönetir ve doğal kaynak aramaları daha verimli yapılabilir.

B. *Yazılımda amaçlananlar şunlardır:*

Yazılımda -daha öncesinden verilen- bir txt linki üzerinden kullanıcı girişiyle satır seçimi yapılmalıdır. Seçilen satırda **Begin (B)** ve **Finish (F)** arasında yer alan koordinat verileri ile şekillerin köşe noktaları hesaplanmalı ve bu durum birden fazla şekilde de aksamadan tekrar edilebilmelidir.

Ardından kullanıcıdan birim platform ve birim sondaj değeri alınmalıdır. Birim platform değeri 1-10 arasında olmalıdır, ancak birim sondaj değerinde böyle bir sınır söz konusu değildir.

Kullanıcıdan veri girişlerinin tamamlanmasının ardından, yine proje dosyasında verilen kurallara bağlı kalarak, gereken matematiksel hesaplamalar yapılmalı ve sonuçlar bir grafik işleme ekranında kullanıcıya sunulmalıdır.

İlk pencerede verilen koordinatlar sonucu oluşan şekiller bir koordinat düzlemi içerisinde gösterilmeli, ikinci pencerede ise bu şekillerin içi optimal parselleme sistemiyle doldurulmalıdır. Her bir parsel boyutu için farklı renkler kullanılmalıdır.

İkinci pencerenin gösterilmesinin ardından kullanıcıya şekil veya şekillerin yüzey alanları gösterilmeli, bu alanlara göre kaynak rezerv değeri bulunmalıdır. Son olarak aynı ekranda kullanıcıya toplam platform sayısı, toplam sondaj sayısı, toplam platform maliyeti, toplam sondaj maliyeti, toplam maliyet ve toplam kâr miktarı gösterilmelidir. Bu şekilde yazılım istenilen tüm amaçları yerine getirmiş olacaktır.

III. Yöntem

Projemiz için -istenildiği gibi- **C** dilini kullandık. Ayrıca bir grafik işleme sisteminin yanı sıra URL işlemek için de bir araca ihtiyacımız vardı. Bu yüzden kullanacağımız sistemler olarak **SDL2** ve **Curl**'ü seçtik, ardından kurulumlarını tamamladık.[1] Gerekli kütüphaneleri de ekledikten sonra kullanıcıdan sırasıyla satır, sondaj ve platform değer girişlerini aldık. Ayrıca bu değerler için aralık belirleyip kontrollerini gerçekleştirdik. Kontroller sonucu elimizdeki değer aralık içerisinde değilse, tekrar kullanıcıdan aynı veriyi almaya yönlendirdik. Bunu yaparken 3 farklı fonksiyonda (SatirAl, SondajAl ve PlatformAl fonksiyonları) **goto** fonksiyonunu kullandık. Bu fonksiyonların yer aldığı kod parçacığını *Deneysel Sonuçlar* bölümünde bulabilirsiniz.

Main içerisinde txt linki için yer açtık ve bu linkten gerekli veri çekme işlemini curl kullanarak gerçekleştirdik. Çektiğimiz linkleri ayırmak ve gereken kısmı almak için bir ayırma fonksiyonu kullandık. Bu fonksiyonun kullanım şekli şöyledir:

1) **UrldenSatiriCek** fonkşyonu ile bir URL'ye HTTP isteği yollanır. Alınan veri (burada metnimiz oluyor) malloc sayesinde bir bellek alanında saklanır.
2) İlgili satırı almak için **SatirAl** fonksiyonu kullanılır. Bu fonksiyon içerisinde kullanıcıya hangi satırdaki veriyi istediği sorulur. Ardından kullanıcının girdiği değere göre ilgili satırdaki veri çekilir. Eğer kullanıcının girdiği satır verisi mevcut veri ile uyuşmuyorsa kullanıcıya yazılı bir uyarı verilir, ardından kullanıcı tekrar satır seçme kısmına yollanır.
3) **strtok** fonksiyonu kullanılarak alınan metin verisi satırlara bölünür. Her satırdaki verileri ayırmak için bizim ayarladığımız bir ayraç kullanılır. Bu ayırma işlemleri **B(,)F** ayraçları kullanılarak gerçekleştirilir.
4) Alınan koordinatlar, yine bizim oluşturduğumuz **koordinatlar** dizisine eklenir. Koordinatlar aynı zamanda bir başka dizide saklanan **siraYap** dizisi ile birleştirilir.

Sonrasında SDL ekranını açmak için gerekli kod parçacıklarını ekledik ve ekran içerisinde kullanılacak grid (ızgara) ve şekil çizgilerinin renk ve tonajlarını belirledik. Bunları yapmak içinse **IzgaraCiz** ve **KoordinatCiz** fonksiyonlarını kullandık. Bunların kullanım şekillerinden de kısaca bahsedecek olursak:

A. IzgaraCiz :

1) **SDL_SetRenderDrawBlendMode** kullanarak gridin saydamlığını müdahale edilebilir hâle getirdik.
2) Grid arka plan ve koordinat çizgisini düzenlemek için renk ve tonaj ayarlamalarını gerçekleştirdik.
3) Fonksiyonun oluşturulken aldığı *satir* ve *sutun* değerleri ile for döngüsüne soktuk. Bu sayede sırasıyla x ye y düzlemlerini çizdik.

İlgili kod parçacığını *Deneysel Sonuçlar* bölümünde bulabilirsiniz.

B. KoordinatCiz:

1) Öncelikle *ilkx* ve *ilky* değişkenlerini koordinat değerlerine sabitledik. Ayrıca bir siraYap[j] değişkeni ekliyoruz. Ardından bir for döngüsü içerisinde (pencerenin boyutlarına uygun olarak) her iki koordinat arasını çizdirdik.
2) Birden fazla şeklin yer alma ihtimalini de göz önünde bulundurduk. Bunu kontrol etmek için de bir if kullandık. Eğer *ilkx* ve *ilky* değerleri sıradaki koordinatlar ile eşitse şeklin başlangıç noktasına geldiğimizi anlıyoruz. Bu yüzden i değerini 2 artırarak yeni şekil ile ilk şekil arasında bir çizgi oluşmasını ve yanış alan hesaplamaları ihtimalini engelliyoruz.
3) Döngü sonucu elimizde kalan güncel j değerini döndürüyoruz, bunu da ileride şekli boyama esnasında kullanıyoruz.

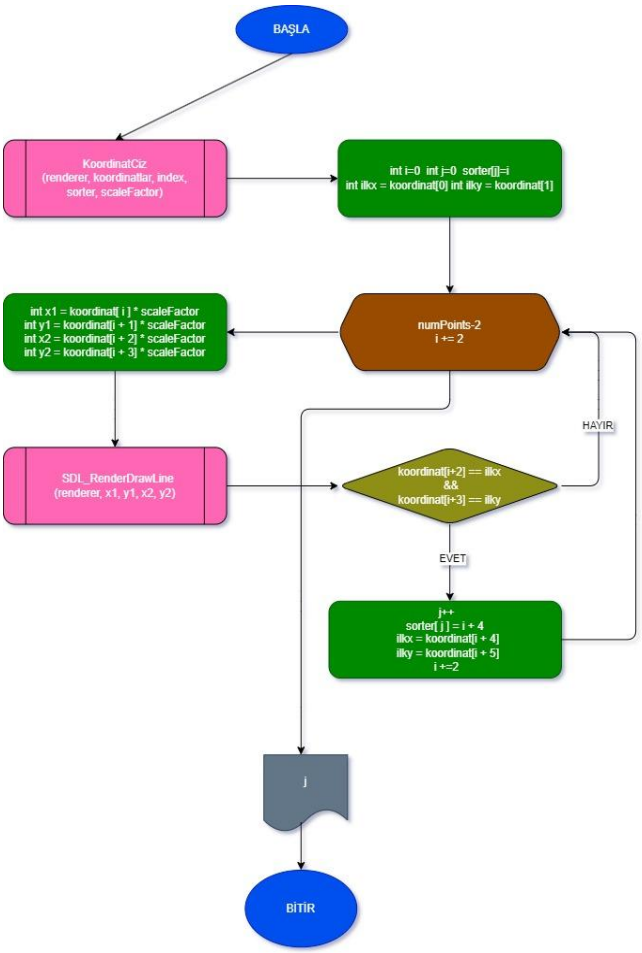
Bu fonksiyonun akış şemasına yansıtılmış halini sayfanın sağ tarafında bulabilirsiniz.

Bunların ardından ilk SDL penceremizi açtık ve koordinat noktalarını kullanarak grid üstüne şekil veya şekillerin çizimlerini gerçekleştirdik. Gözle rahat görülebilir bir değer olması için boyutu 10 olarak ayarladık.

İlk pencerenin kapatılması ardından ikinci pencereyi açtık. Yine grid ve boyama renklerini ve tonajlarını ayarladık. Bu pencerede de şekil veya şekillerin içlerini doldurduk. Bunu yaparken birimkarelerin şeklin veya şekillerin içinde veya şekille temas halinde olup olmadığını kontrol eden bir sistem olan ***Ray Casting*** sistemini kullandık. İlgili sistem hakkında bizim de yararlandığımız linkleri kaynakçaya ekledik.[2][3]

Şekilleri doldururken sunumda da göreceğiniz üzere tam birimkareli bir boyama yönteminden ziyade daha yuvarlak köşeli bir boyama stili kullandık. Bunun en önemli sebeplerinden birisi, tam birimkareli bir sistem isterlerinize daha uygun olsa da bu sistemde alan hesaplama konusunda hataların oluşması. Bu tür alan hesaplama işlemlerinin daha tutarlı olması adına birimkareli sistemden feragat ettik.

Şekillerin doldurulmasının ardından alanlarını hesapladık ve kaynak rezerv alan değerini bulduk. Girilen sondaj ve platform değerlerini kullanarak optimal hesaplamaları gerçekleştirdik. Bu değerleri cmd ekranına yazdırmamızın ardından da kodumuzu sonlandırdık.



Koordinatları Çizme Algoritmasına ait bir akış şeması

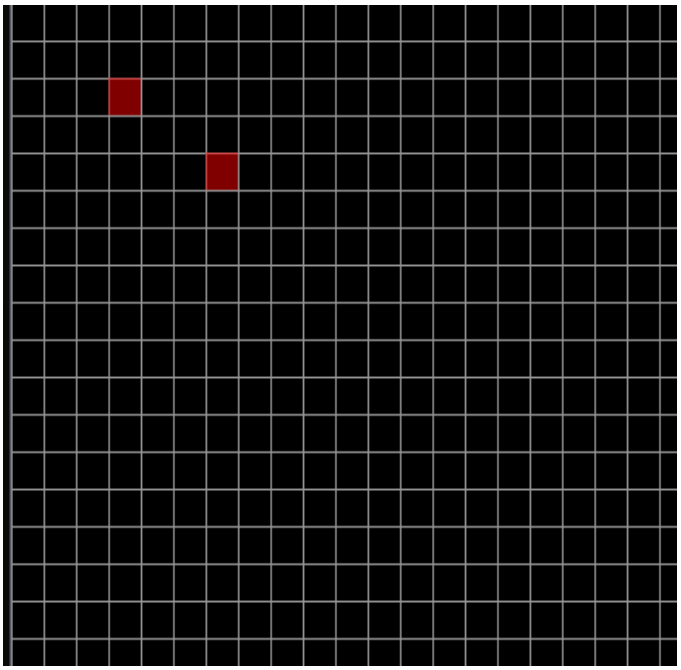
IV. DENEYSEL SONUÇLAR

```
void IzgaraCiz(SDL_Renderer *renderer, int satir, int sutun, int boyut)
{
    // Grid çizmek için döngüler
    SDL_SetRenderDrawBlendMode(renderer, SDL_BLENDMODE_BLEND);
    SDL_SetRenderDrawColor(renderer, 150, 150, 150, 100);

    for (int i = 0; i <= satir; i++)
    {
        SDL_RenderDrawLine(renderer, 0, i * boyut, sutun * boyut, i * boyut);
    }

    for (int i = 0; i <= sutun; i++)
    {
        SDL_RenderDrawLine(renderer, i * boyut, 0, i * boyut, satir * boyut);
    }
}
```

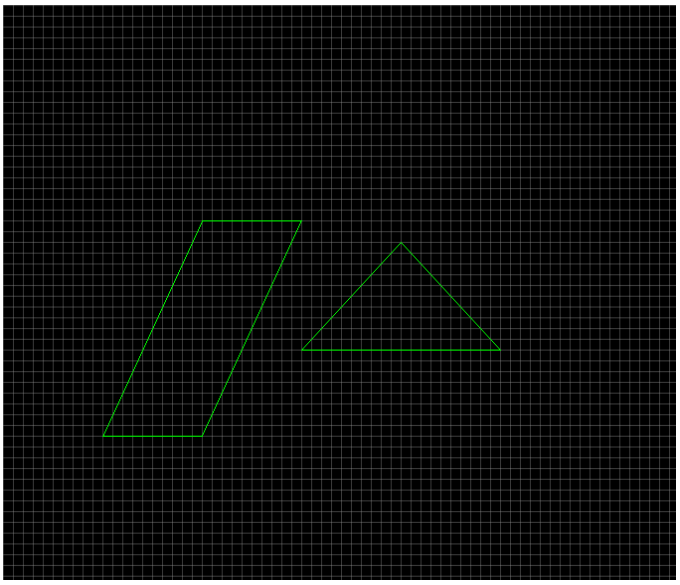
Grid (Izgara) çizmek için kullandığımız fonksiyon örneği

```c
int SatirAl(int satir)
{
bas:
    printf("Kacinci satirdaki koordinatlari cizelim?:");
    scanf("%d",&satir);
    if(satir <= 0)
    {
        printf("Lutfen gecerli bir sayi giriniz.\n");
        goto bas;
    }
    return satir;

}

int SondajAl(int sondaj)
{
bas:
    printf("Birim sondaj maliyeti ne kadar olacaktir?:");
    scanf("%d",&sondaj);
    if(10 < sondaj || sondaj< 1)
    {
        printf("Lutfen gecerli bir sondaj maliyeti birimi giriniz.\n");
        goto bas;
    }
    return sondaj;

}

int PlatformAl(int platform)
{
bas:
    printf("Birim platform maliyeti ne kadar olacaktir?:");
    scanf("%d",&platform);
    if(platform <= 0)
    {
        printf("Lutfen gecerli bir platform maliyeti birimi giriniz.\n");
        goto bas;
    }
    return platform;

}
```

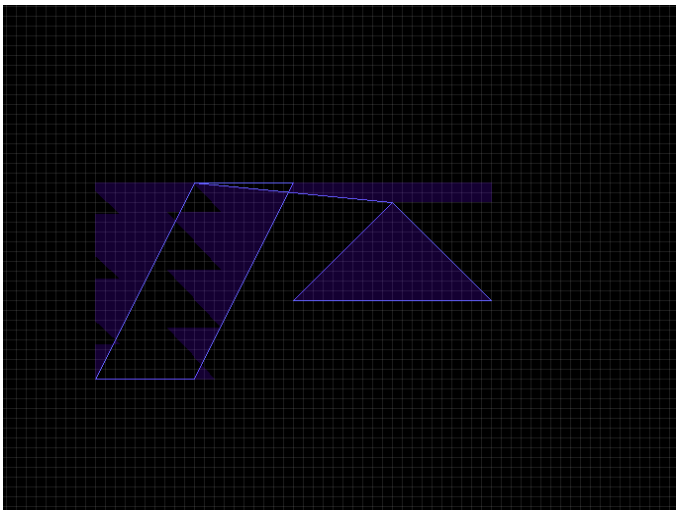
Satır, Sondaj ve Platform değerlerini aldığımız ve kontrol ettiğimiz kod parçacığı



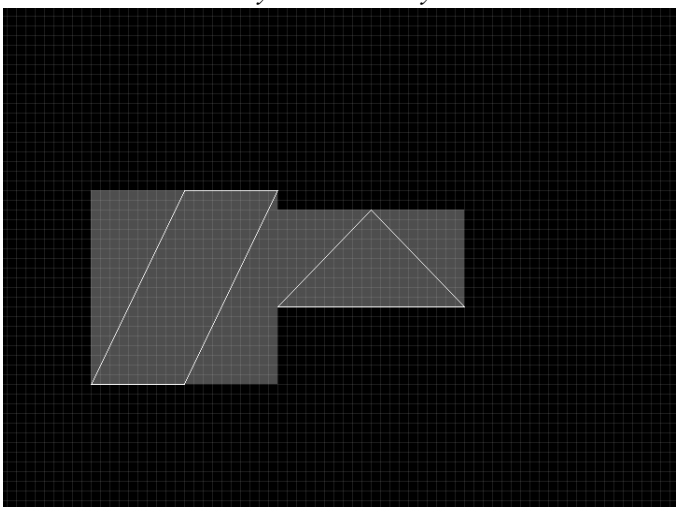
Şekillerde belirli noktaları boyamak için yaptığımız ilk örnek



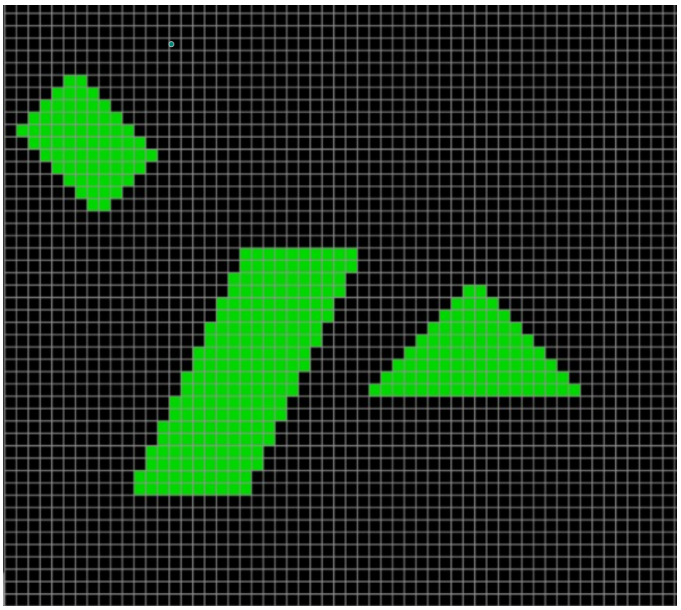
İlk link üzerinden 2. satır seçilmesi sonucu oluşan 1. pencere



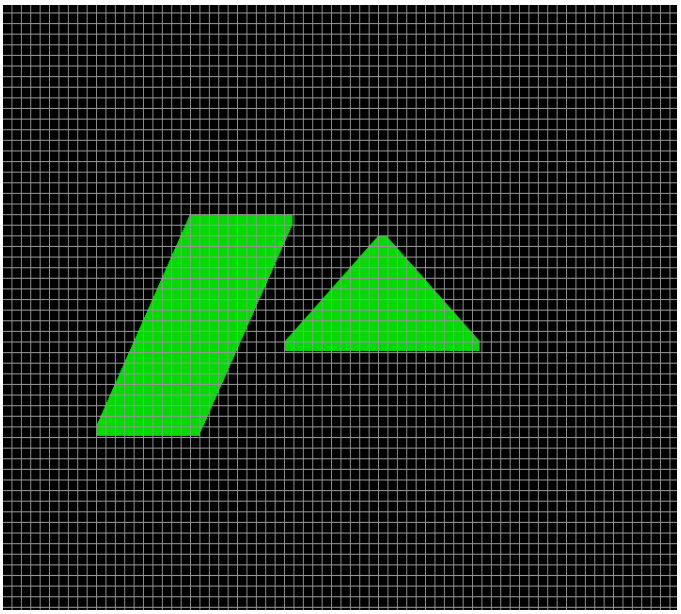
Güncel olmayan bir alan boyama denemesi



Bir başka güncel olmayan alan boyama denemesi



Yanlış alan değerleri aldığımız birimkare sisteminden bir ekran görüntüsü



İlk pencerenin kapanmasının ardından alan boyaması yapılmış 2. pencere

V. Sonuç

Bu projenin sonucunda linkten veri çekip kullanmayı, bu verilerle hesaplamalar yapmayı öğrendik. Grafik işleme sistemleri hakkında deneyim sahibi olduk. Göze oldukça basit gözüken bazı grafik işlemlerinin ne kadar zorlayıcı olabildiklerini fark ettik.

Bunların yanı sıra takım halinde bir proje yapmanın kolaylıklarını ve zorluklarını ilk elden deneyimledik. Birbirimizin eksik yönlerini geliştirmenin ve sürekli olarak aktif kalmanın yollarını öğrendik. Tam olarak istediğimizi elde edemesek de, her işin tam olarak istediğimiz gibi gidemeyeceğini gördük. Bizce en önemlisi, zamanı verimli kullanmamız gerektiğini ve bunu nasıl yapmamız gerektiğini, proje sürecinde iletişimimizin nasıl olması gerektiğini öğrendik.

Bilgilendirme

Projemiz esnasında Abdulrahman Dülek'in dosyasında int WinMain, Anıl Can Göl'ün dosyasında int main kullanılmıştır. Sebebi ise Abdulrahman Dülek'in dosyasının int main ile çalıştırma denendiğinde WinMain'e referans aramasıdır. Bundna kaynaklı tek satırlık bir kod farkı söz konusudur.

Ayrıca raporda isterlerin tamamı başarıyla tamamlanmış olarak yazılmasına rağmen, parselleme konusunda yaşadığımız sıkıntılar sonucu optimal alan hesaplamamız mümkün olmamıştır. Bunun yerine elimizdeki mevcut alan verileri üzerinden tekil hesaplamalar yapılmış ve onların sonuçları ekrana yansıtılmıştır.

Kaynaklar

[1] Codeincodeblock.com. "*Setup SDL 2 in Codeblocks*". 16.11.23. https://www.codeincodeblock.com/2011/02/setup-sdl-12-version-in-codeblock.html .

[2] Youtube.com. "*Checking if a point is inside a polygon is RIDICULOUSLY simple (ray casting algorithm) - inside code*". 28.10.23. https://www.youtube.com/watch?v=RSXM9bgqxJM .

[3] Wikipedia.org. "*Ray Casting*". 30.10.23. https://en.wikipedia.org/wiki/Ray_casting.