Programlama Laboratuvarı I
Proje I
Bilgisayar Mühendisliği Bölümü
Kocaeli Üniversitesi

Sümeyra USTA
Bilgisayar Mühendisliği(İÖ)
Kocaeli Üniversitesi
220202070

I. ÖZET

Bu rapor Programlama Laboratuvarı I dersinin 1.projesi için oluşturulmuştur. Bu proje C dili kullanılarak geliştirilmiştir. Projeyi geliştirme ortamı olarak Visual Studio Code kullanılmıştır.

Belgede akış diyagramı , özet, giriş, yöntem ,deneysel sonuçlar gibi projeyi açıklayan başlıklara yer verilmiştir. Belge sonunda projenin sonucu ve projeyi hazırlarken kullanılan kaynaklar bulunmaktadır.

II. GİRİŞ

Bu rapor, C dilinde algoritma geliştirilen bir projenin detaylarını içermektedir. Projenin amacı sondaj şirketinin deniz kaynaklarını araştırma ve çıkarma işlemlerini en kârlı şekilde nasıl bölümleyeceğini belirleyen bir yazılım geliştirmek. Bu süreçte, kullanıcı tarafından alınan bilgiler sonucu hesaplamalar yapılmıştır. Ardından hesaplanan sonuçlar iki pencere halinde sunucuya sunulmuştur.

İlk adımda, verilen linkteki verilerin kullanıcının isteği üzerine işlenmiştir. Linkden okunan veriler ilk olarak hepsini barındıran bir dosyaya yazdırılmıştır. Tüm linkin okumasının yapıldığı dosyadan kullanıcının girdiği ister üzerine başka bir dosyaya, dosya işlemleri ile yazdırılmıştır. Ardından koordinat belirten noktalar bulunarak kaydedilmiştir.

İkinci adımda kullanıcının istediklerini barındıran dizi, kontrol edilerek koordinatların kaç adet şekil oluşturduğu ve şekillerin köşe sayısı bulunarak tutulurak gösterilmiştir.

Ardından SDL kullanılarak verilen şekiller önce ışın kontrol yöntemi ile üçgensel parçalar halinde kontrol edilerek üçgenin sınırında ya da içinde olan kareler birim karelerden oluşan karelerle boyanmıştır. Alınan çıktı kullanıcıya sunulmuştur.

Koordinatların sahip olduğu renklerle kontrol sağlayarak ve komşu kuralına dikkat edilerek optimum alan-kare durumuna hazırlanmıştır. Alınan çıktı kullanıcıya sunulmuştur.

Son adımda son verilerden renk kontrolü yapılarak kar-zarar durumu hesabı yapılmıştır ve sonuç kullanıcıya sunulmuştur.

III. YÖNTEM

A. *CURL Kütüphanesi ile Veri Alımı*

Program, CURL türünde bir işaretçi oluşturarak başlar. CURL kütüphanesi, ağ işlemleri için standart bir arayüz sağlar. ‘curl_global_init(CURL_GLOBAL_DEFAULT)’ çağrısı ile kütüphane, varsayılan ayarlarla başlatılır. ‘curl_easy_init()’ fonksiyonu, bir CURL oturumu başlatmak için kullanılır.

CURL tarafından alınan verilerin yazılacağı "urldenOkunan.txt" adlı bir dosya, yazma modunda ("wb") açılır. Dosya açılamazsa, bir hata mesajı yazdırılır ve program 1 değeri döndürerek sonlanır.

‘CURLOPT_URL’ seçeneği, verilerin alınacağı URL adresini belirtmek için kullanılır. Bu durumda, "http://bilgisayar.kocaeli.edu.tr/prolab1/prolab1.txt" adresidir.’ CURLOPT_WRITEFUNCTION ‘ , indirilen verilerin işlenmesi için kullanılacak geri çağırım fonksiyonunu (yazdir) belirtir.

‘curl_easy_perform(curl)’ çağrısı ile HTTP GET işlemi başlatılır ve belirtilen URL'den veriler alınır.

Eğer ‘curl_easy_perform ‘işlemi başarısız olursa, hata mesajı yazdırılır. Başarılı olursa, alınan veriler daha önce açılmış olan dosyaya yazılır.

İşlem tamamlandıktan sonra,' fclose(fp)' çağrısı ile dosya kapatılır ve 'curl_easy_cleanup(curl)' fonksiyonu ile CURL oturumu temizlenir.

Kullanıcıdan, işlem yapmak istediği satır numarası girdisi alınır.

'target_row' fonksiyonu, kaynak dosyadan ("output.txt") belirtilen satırı alıp, hedef dosyaya ("kullanicinin_istedigi.txt") kopyalar.

'separate_coordinates' fonksiyonu, hedef dosyadaki verileri işler ve gerekli koordinat bilgilerini ayrıştırır.

B. Şekil Bulma

Bu çalışmada, belirli bir geometrik yapıyı tanımlamak ve ayırt etmek için bir dizi köşe noktası üzerinde işlem yapan 'find_shape 'adında bir C fonksiyonu geliştirilmiştir. Fonksiyon, üç farklı şekli tanımlayacak şekilde parametreleştirilmiş ve her bir şekil için başlangıç koordinatlarına geri dönene kadar veya noktalar tükenene kadar döngüsel bir kontrol mekanizması içermektedir. Fonksiyonun parametreleri arasında köşe noktalarını içeren bir dizi, bu noktaların sayısını ve her bir tanımlanacak şekil için bir dizi ve bu dizilerin indekslerini tutacak göstergeler yer almaktadır.

Fonksiyon ilk olarak dizideki ilk noktadan başlayarak, başlangıç noktasına geri dönene kadar noktaları ilk şekil dizisine ekler ve bu süreci tüm noktalar tükenene kadar devam ettirir.

İlk şeklin tamamlanmasıyla, ilgili indeks güncellenir ve eğer daha fazla nokta kalmışsa, ikinci şekil tanımlama sürecine geçilir.

İkinci şekil için de benzer bir prosedür izlenir ve uygun şekilde indeks güncellenir. Ancak, ikinci şekil için yeterli nokta bulunmazsa, ilgili şekil dizisinin indeksi sıfırlanır.

Üçüncü şekil için de aynı işlem uygulanır ve bir hata düzeltmesi yapılır: eğer 'sekil3Index' iki noktadan az ise, bu durum bir şekil oluşturulamayacağı anlamına gelir ve indeks sıfırlanır. Bu yöntem, veri setindeki noktaların sıralı bir şekilde farklı geometrik şekiller oluşturup oluşturmadığını kontrol eder ve sonuçları ilgili dizilere kaydeder. Fonksiyonun bu özel yapısı, şekil tanıma ve geometrik veri işleme gibi alanlarda geniş uygulama olanakları sunmaktadır.

C. SDL ile Pencere İşlemleri

Uygulamanın başlangıcında, 'SDL_Init' fonksiyonu çağrılarak SDL kütüphanesi başlatılır. Eğer SDL başlatma işleminde bir hata oluşursa, bir hata mesajı basılır ve uygulama false değeri dönerek sonlandırılır. İki adet pencere 'window1 ve window2' ve bunlara ait çizim işlemleri için iki adet grafik işleyici 'renderer1 ve renderer2' oluşturulur. Her pencere için, pikselleri saklamak üzere bir grafik tamponu 'framebuffer' ayrılır ve bu tamponlar üzerinde çizim yapmak için SDL dokuları 'framebuffer_texture1 ve framebuffer_texture2' yaratılır.

'draw_pixel' fonksiyonu, belirli bir pencerenin grafik tamponuna bir piksel çizmek için kullanılır. Pikselin koordinatları ve rengi parametre olarak alınır. Geçersiz koordinatlar göz ardı edilir. 'render_framebuffer' fonksiyonu, bir pencerenin grafik tamponundaki içeriği ekrana yansıtmak için kullanılır. Bu işlev, grafik tamponunu günceller ve SDL dokusunu kullanarak pikselleri ekranda gösterir.

'clear_framebuffer' fonksiyonu, bir pencerenin grafik tamponunu belirli bir renkle doldurmak için kullanılır. 'clear_screen' fonksiyonu, bir pencerenin içeriğini temizler. 'destroy_window' fonksiyonu, programın sonunda oluşturulan pencereleri ve grafik işleyicileri kapatmak ve ayrılan belleği serbest bırakmak için kullanılır.

'fix_framerate' fonksiyonu, uygulamanın kare hızını sabitlemek için kullanılır. Bu fonksiyon, her kare arasında belirli bir süre gecikme ekleyerek uygulamanın çok hızlı çalışmasını önler ve daha düzgün bir animasyon sağlar.

'get_framebuffer' fonksiyonu, belirli bir pencereye ait grafik tamponunu almak için kullanılır. Bu fonksiyon, programın farklı bölümlerinden ilgili grafik tamponuna erişim sağlar.

'fill_polygon' fonksiyonu, bir poligonun içini belirli bir renk ile doldurmak için kullanılır. İlk olarak, poligonun sınırlayıcı kutusunu (bounding box) hesaplamak üzere poligonun köşeleri taranır ve en küçük ve en büyük x ve y koordinatları belirlenir.

Daha sonra, bu sınırlayıcı kutu içindeki her nokta için, noktanın poligonun içinde olup olmadığı 'is_point_inside_polygon' fonksiyonu kullanılarak kontrol edilir. Eğer nokta poligonun içindeyse, 'draw_pixel' fonksiyonu çağrılarak ilgili nokta belirtilen renkle boyanır ve boyanan piksel sayısı artırılır.

Bu yöntemle, poligonun tüm iç alanı yinelemeli olarak taranır ve her bir iç nokta için renklendirme işlemi yapılır. İşlem sonucunda, boyanan toplam piksel sayısı geri döndürülür.

D. Koordinatlarla Şekil Çizme ve İçini Doldurma

'is_point_inside_polygon' fonksiyonu, bir noktanın bir poligonun içinde olup olmadığını belirlemek için kullanılan bir yöntemdir. Bu yöntem, bir poligonun kenarları boyunca yatay bir ışın çizerek ve bu ışının poligonun kenarlarıyla kaç kez kesiştiğini sayarak çalışır. Kesişim sayısı tek ise, nokta poligonun içindedir; çift ise dışındadır.

Fonksiyon, öncelikle poligonun her kenarı için döngüye girer. Her iterasyonda, poligonun iki ardışık köşesi alınır ve bu köşeler arasında bir kenar oluşturulur. Eğer bu kenar yatay ise ve nokta da bu kenarın yüksekliğindeyse, fonksiyon true döndürerek noktanın poligonun üzerinde olduğunu belirtir. Eğer kenar yatay değilse, kenarın ışın ile kesişip kesişmediği kontrol edilir. Kesişim noktasının x koordinatı hesaplanır ve eğer bu koordinat, incelenen noktanın x koordinatından büyükse, kesişim sayısı artırılır.

E. *Optimum Alan-Kare*

Çalışmamız, grafik tamponunun kopyalanması, belirli renk desenlerinin algılanması ve renk dönüşümü işlemlerini içermektedir. İlk olarak, copy_framebuffer işlevi, belirli bir pencerenin mevcut grafik tamponunun içeriğini bir yedekleme dizisine kopyalar. Bu işlem, sonraki renk algılama ve dönüşüm işlemlerinde bir referans noktası olarak kullanılır.

Renk algılama işlevleri, is_2x2_square_green, is_4x4_square_blue, is_4x4_square_red ve is_16x16_square_yellow şeklinde tanımlanmıştır. Her bir işlev, belirli bir boyutta karenin tamamının tek bir renk olup olmadığını kontrol eder. Bu işlevler, grafik tampon üzerindeki belirli bir pozisyonda başlayarak, her bir pikselin renk değerini belirtilen renk ile karşılaştırır. Eğer tüm pikseller beklenen renk ise, işlev true değeri döndürerek karenin tamamının belirli bir renkte olduğunu doğrular.

Renk dönüşüm işlevleri, 'convert_2x2_green_to_blue', 'convert_4x4_blue_to_red', 'convert_4x4_red_to_8x8_yellow' ve 'convert_8x8_yellow_to_16x16_pink' olarak tanımlanmıştır. Her bir işlev, belirli bir renkteki kareleri algılar ve bu karelerin renklerini başka bir renge dönüştürür. Örneğin,' convert_2x2_green_to_blue' işlevi, tümü yeşil olan 2x2 boyutundaki kareleri mavi renge dönüştürür. Renk dönüşümü, önce karenin boyutuna göre bir algılama işlemi yaparak ve ardından tespit edilen her pikseli yeni renge boyayarak gerçekleştirilir. Boyama işlemi, 'draw_pixel' işlevi kullanılarak yapılmaktadır.

'are_neighbors_not_colored' işlevi, belirli bir karenin komşu piksellerinin belirli renklerde olup olmadığını kontrol eder. Bu işlev, belirtilen karenin çevresindeki pikselleri dikkatlice inceler ve eğer herhangi bir komşu piksel belirtilen renkler dışında bir renkteyse, işlev false değeri döndürerek renk dönüşümü için uygun olmadığını belirtir. Bu kontrol, renk dönüşüm işlevlerinde bir ön koşul olarak kullanılır.

IV. DENEYSEL SONUÇLAR

Projede belirlenen yöntemlerin uygulanmasının ardından, sonuçlar iki ayrı pencere üzerinde kullanıcıya sunulmuştur. İlk pencerede, CURL ile alınan veriler işlenmiş ve kullanıcı tarafından belirtilen satır numarasına göre filtrelenmiştir. Elde edilen veriler, koordinat sistemine dönüştürülerek çeşitli geometrik şekillerin tanımlanması sağlanmıştır. Bu işlem, kullanıcının veri seti üzerindeki belirli bir bölümü detaylı bir şekilde incelemesine olanak tanımıştır.

İkinci pencerede ise SDL kullanılarak şekillerin grafiksel temsili gerçekleştirilmiştir. İlk olarak, ışın kesişim yöntemi ile şekillerin sınırları belirlenmiş ve bu sınırlar içerisinde kalan pikseller uygun renklerle boyanarak şekiller görselleştirilmiştir. Daha sonra komşuluk kuralına uygun bir şekilde renk dönüşümleri yapılmış ve böylece şekillerin farklı renklerle işaretlenmesi sağlanmıştır. Bu işlemler, kullanıcının şekillerin boyutlarını ve konumlarını daha iyi anlamasına yardımcı olmuştur.

Sonuç olarak, kullanıcının istediği veri üzerinde yapılan analizler ve görselleştirmeler, son derece etkili ve verimli bir şekilde gerçekleştirilmiştir. Kullanıcı, projede geliştirilen araçların yardımıyla, deniz kaynaklarını araştırma ve çıkarma işlemlerini en kârlı şekilde nasıl planlayabileceğini anlamıştır. Grafik tamponunun manipülasyonu ve renk dönüşüm algoritmaları, kullanıcının optimum alan-kare durumunu tespit etmesine imkan tanımış ve böylece kar-zarar durumu hesaplamaları için gerekli veri sağlanmıştır.

V. SONUÇ

Bu proje, kompleks veri setlerinin işlenmesi ve görselleştirilmesi üzerine odaklanmaktadır. Geliştirilen yazılım, kullanıcıya veri analizi ve şekil tanıma konularında güçlü araçlar sunarak, deniz kaynaklarının araştırılması ve çıkarılması gibi spesifik görevler için etkili bir çözüm oluşturmuştur. Proje boyunca karşılaşılan zorluklar, sistematik problem çözme yöntemleri ve yaratıcı algoritma tasarımları sayesinde aşılmıştır. Elde edilen sonuçlar, projenin hem teorik hem de pratik beklentileri karşıladığını göstermektedir.

Son olarak, bu rapor, projenin nasıl gerçekleştirildiği ve hangi yöntemlerin kullanıldığı hakkında kapsamlı bir rehber sunmaktadır. Raporun sonunda, projenin geliştirilmesinde kullanılan literatür ve kaynaklara yer verilmiştir. Bu kaynaklar, projenin daha iyi anlaşılmasını sağlamakta ve bu alanda çalışacak diğer araştırmacılar için bir başvuru kaynağı oluşturmaktadır.

KAYNAKLAR

[1] *"cURL Command in Linux with Examples - GeeksforGeeks," GeeksforGeeks, [Online]. Available: https://www.geeksforgeeks.org/curl-command-in-linux-with-examples/.*

[2] *"Curl Command With Examples - Hostinger Tutorials," Hostinger, [Online]. Available: https://www.hostinger.com/tutorials/curl-command-with-examples-linux/.*

[3] *"Curl Command Tutorial - YouTube," YouTube, [Online]. Available: https://www.youtube.com/watch?v=7XUibDYw4mc.*

[4] *"SDL2 Tutorials - SDL Wiki," SDL Wiki, [Online]. Available: https://wiki.libsdl.org/SDL2/Tutorials.*

[5] *"SDL 2.0 Tutorial for Beginners - YouTube," YouTube, [Online]. Available: https://www.youtube.com/watch?v=XfZ6WrV5Z7Y.*

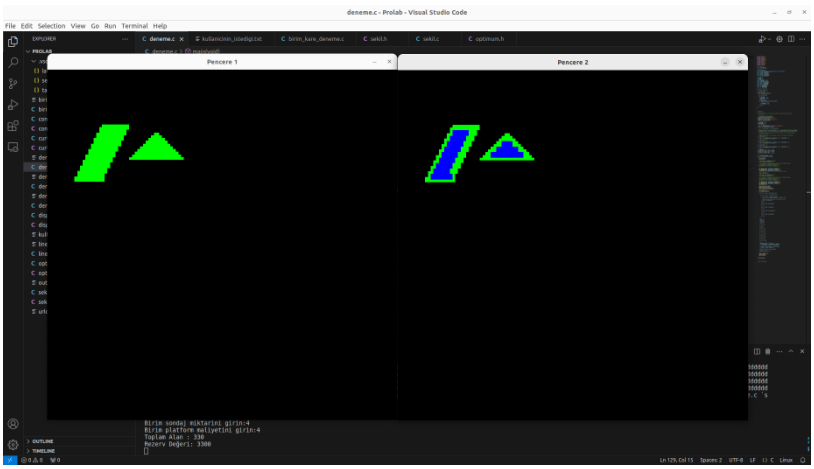
[6] *"SDL Library in C/C++ with Examples - GeeksforGeeks," GeeksforGeeks, [Online]. Available: https://www.geeksforgeeks.org/sdl-library-in-c-c-with-examples/.*

[7] *"Learn SDL2 - Strivix," Strivix, [Online]. Available: https://www.strivix.com/learn-programming/learn-sdl/learn-sdl2.*

[8] "C Game Development with SDL2": https://www.youtube.com/watch?v=rR7xWyUWcBA

[9] "C Standard Library Functions":https://www.programiz.com/c-programming/library-function

Ek1: Akış Diyagramı



Ek2: Kullanıcının ekranı