GALATASARAY ÜNİVERSİTESİ VERİ YAPILARI VE İLERİ BİLGİSAYAR PROGRAMLAMA PROJE RAPORU

Mayıs-2021

Deniz TÜRK

Fatih Halid KARAER

İÇERİK

BÖLÜM 1

GİRİŞ

Projenin Amacı

Projenin temel amacı veri yapıları ve algoritmalara giriş dersinde öğrenilen teknikleri uygulayarak görüntü işleme ve manipülasyon tekniklerine ,bunların bir uygulamasını yaparak giriş yapmak. Bu proje Günlük hayatta sürekli iç içe olduğumuz ve görsel olarak ,insanlar için bir anlam ifade edebilen ve bir iletişim aracı olan fotoğraf dosyalarının arkasında taşıdığı matematiği giriş seviyesinde anlayabilmek. Fotoğraf dosyalarının taşıdığı dijital bilgiyi okuyabilmek, ağlar üzerinden insanların birbirine gönderebildiği, saklayabildiği bu dijital bilginin nasıl veya hangi formatlarda taşındığını veya taşınabileceğini öğrenmek ,bu datayı farklı veri yapılarına aktarıp yönetebilmek, çeşitli senaryolara göre uygun veri yapısı analizini yapabilmek, çeşitli senaryolara göre uygun algoritmaları seçebilmek ve analiz yapabilmek, çalışılacak datayı analiz edip daha sonra bu analize göre uygun araçları (tool) seçebilmek ve data üzerinde manipülasyonlar yapmayı kapsar.

Bütün bu konuları ele almak için, girdi olarak verilen fotoğraf dosyaları üzerinde c yazılım dilini kullanarak yukarıda açıklanan objektifler doğrultusunda işlemler yapacağız. Bu işlemler her bir fotoğraf dosyasını tek tek okuyup parlaklık, kontrast, terse çevirme, keskinlik gibi ve benzeri temel fotoğraf efektlerini uygulayabilmek ayrıca bir nesneyi kapsayan birden çok fotoğrafı girdi olarak alarak median filtreleme tekniğiyle bir turisti fotoğraftan kaldırdığımız tek bir fotoğraf elde etme üzerine olacaktır.

Gerçek Hayatta Nasıl Kullanım Alanları ve Görüntü İşleme Teknolojisinin Örnekleri

Günümüzde görüntü işleme birçok alanda kullanılıyor. Otomasyon sistemlerinin hayatımıza giderek daha fazla girmesiyle birlikte birçok sektörde örneklerine rastlamak mümkün. Askeri alandan örnek vermek gerekirse insansız hava araçlarının kaydettiği görüntüleri yapay zekayı da dahil ederek düşman birimlerinin sığınak , mühimmat deposu gibi bölgelerini bilgisayar yardımı ile belirleyebilmek için kullanılıyor, denizaltılarda kullanılan sualtı görüntüleme sistemleri ise farklı bir uygulama alanı . Günlük hayatta ise özellikle elektrikli arabalar ile birlikte yaygınlaşan özelliklerden olan oto sürüş sistemleri örnek verilebilir. Bu sistemlerde yol ve şeridi takip edebilme ve olası kaza durumlarını öngörebilmek için sensörlerle birlikte görüntü işleme teknikleri de kullanılıyor. Bu uygulamalar görüntü işleme için örnek teşkil etse de bu projeye en çok yaklaşan örnekler için sinema endüstrisinde sıklıkla kullanılan ve özellikle sosyal medya platformlarının da çok fazla yaygınlaşması ile birlikte kullanım alanları hızlı bir şekilde artan photoshop, premiere pro ,after efects,Unity, Unreal Engine gibi yazılımları gösterebiliriz. Her ne kadar bu programlar dinamik yapıda olsalar da, (bu programlar kullanıcılarından kod yazmalarını beklemiyorlar fakat kullanıcının program üzerindeki seçenekleri kullanarak sınırsız kombinasyonda işlemler yapabiliyorlar), Temel olarak aynı prensiplere ve tekniklere dayanır. Örneğin bu projede uygulanan turist silme uygulamasını belirtilen programlarda kullanıcılar layer'lar yardımıyla istenilen alanlara "masking" işlemleri ile yapabilirlerken, bu projede pixellerdeki datalar karşılaştırılarak turist kaldırma işlemi otomatik olarak yapılmıştır.

Son örnek olarak Epic Games firmasının bir ürünü olan Unreal Engin'i ele alacağız. Aslında bir oyun motoru olarak tasarlanan Unreal Engine yazılımı son zamanlarda giderek artan bir hızda film ve sinema endüstrisi için teknolojiler geliştirmeye devam ediyor . Oyun ve sinema endüstrisi bu teknolojiler ile iç içe geçmeye başladı bile. Görüntü işleme teknikleri ve teknolojileri için ,bu yazılım en uç noktalardan biridir. Grafik işlemlerinde ,3d hesaplamalarda, vfx ve render işlemlerinde kullanılan donanım olarak cpu dan gpu ya doğru bir geçişin, gpu'ların işlem kapasitelerinin hızlı bir şekilde artmasıyla mevcut olduğunu biliyoruz.

Hali hazırda gpu'yu çok verimli kullanan bu oyun motorunun görüntü işleme üzerine çok güzel bir örnek olacak ve sinema endüstrisinde green screen teknolojisini ortadan kaldırarak devrim yaratacak yeni bir teknolojisine değineceğiz. Bu teknoloji her tarafı yeşil film setlerine artık ihtiyaç kalmamasını sağlayan ve muhtemelen zaman içerisinde maliyetlerinin de ucuzlamasıyla giderek sektörde daha fazla yer alacak bir teknoloji. Setlere kurulan kavisli ve devasa led ekranlar kullanılarak, unreal engine ile senkronize olan ve kameralara takılan sensörler yardımıyla **real time render** yapabilen unreal engine, kameraya senkronize şekilde devasa televizyonlara real time görüntü yerleştirmesi yapabiliyor .Bu sayede post production aşamasına gerek kalmadan efektler gerçek zamanlı olarak ve yönetmenin direkt olarak uygulanmış halini görebileceği bir şekilde uygulanabiliyor .Görüntü işlemenin uygulamalarından biri olan bu teknoloji, filmlerdeki vfx teknolojisinin gidişatını tamamen değiştirebilecek potansiyele sahip.

Projede Kullanılan Teknoloji

Projede teknoloji olarak PGM dosya formatı kullanılmıştır bu tercihin nedenine gelmeden önce PGM dosya formatının özelliklerinden ve tarihinden bahsedeceğiz. PGM (Portable Gray Map) dosya sistemi 1988'de yayınlanan Netpbm projesinin bir parçasıdır. Dolayısıyla bu projeye ve bu proje kapsamında geliştirilen ve PGM dosya formatlarına çok benzer diğer dosya formatlarına da kısaca değineceğiz. Netpbm (eski adıyla Pbmplus), açık kaynaklı bir grafik programları paketi ve bir programlama kitaplığıdır.

Temel olarak Unix dünyasında kullanılır, burada tüm büyük açık kaynaklı işletim sistemi dağıtımlarında bulunur, ancak aynı zamanda Microsoft Windows, macOS ve diğer işletim sistemlerinde de çalışır. Netpbm projesi tarafından çeşitli grafik biçimleri kullanılır ve tanımlanır. Portable pixmap format (PPM) sadece siyah ve beyaz renklerin olduğu , Portable graymap format (PGM) bizim projede kullancağımız format ve rgb kanallarına da sahip olan portable bitmap format (PBM) bunlardan bazıları. Bunlara bazen toplu olarak taşınabilir anymap formatı (PNM) olarak da atıfta bulunulur.

Bu fotoğraf formatları platformlar arasında kolaylıkla değiş tokuş edilmek amacıyla tasarlanmış formatlarıdır. Kolaylıkla transfer edilebilmesinin sebebi ise çok basit bir dosya formatı olmasıdır ki bu projede kullanılmasının en önemli sebeplerinden bir tanesidir (Bu teknolojinin kullanılmasının ayrıntılı sebeplerine 2. Bölümde daha geniş ve teknik açıdan değinilecektir).

Projede C yazılım dili kullanılmış olup, proje linux işletim sisteminde yazılıp derlenmiştir. Projenin derlenip çalıştırlabilmesi için makefile inşa dosyasına ihtiyaç vardır ve proje dosyaları arasında mevcuttur.

Çözüm ve Diğer Çözüm Alternatifleri Arasındaki Farklar

Günlük hayatta kullanılan diğer görüntü işleme çözümlerde özellikle yaratıcılığa dayanan işlemlerde programlar daha dinamik ve kullanıcıya çok daha fazla alternatif araçlar sunuyor. Sanatsal sektörlerde kullanıcıya daha fazla seçenek sunulması da zaten çok normal. Fotoğrafları ve videoları işleyip görsel anlamda insanların beğenisini kazanabileceği işler ortaya koymak en azından günümüzde hala bilgisayar ve insan ortaklığına kalıyor. Fakat bu projedeki işlemlerin direkt olarak bilgisayarlara yaptırıldığını düşündüğümüzde, projedeki uygulamaların sunduğu çözümler şunlar olabilir. Bir şirketin, uygulamasında kullanıcılarından spesifik bir konu için veri tabanında saklayabileceği fotoğraflar isteyebilir. Bu fotoğraflar üzerinde otomatik olarak bazı doğrulamaların yapılması gerekliliğinde gelecek fotoğrafların neleri içereceği zaten belli ise sürecin hızlandırılması için otomatik doğrulama işlemleri gibi işlemler bilgisayarlara yaptırılabilir. Örnek olarak bu projede kullanılan median filtreleme işleminde bütün fotoğraflarda turistin olduğunu ve aynı kadrajda çekilen fotoğraflardan oluştuğunu biliyorsanız median filtreleme ile turisti kaldırabilirsiniz fakat kullandığınız fotoğraflar birbirinden alakasız farklı fotoğraflar olursa çok farklı sonuçlar alırsınız. Bu projenin sunduğu çözümler daha çok 2. senaryoya hitap ediyor diyebiliriz.

BÖLÜM 2

PROJE MİMARİSİ

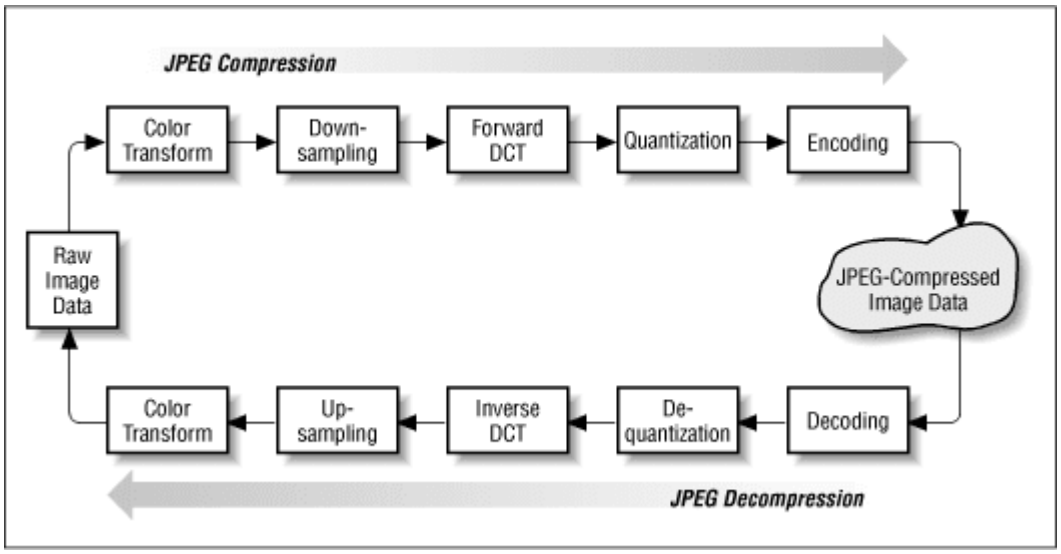
Teknoloji Olarak Neden PGM Formatı Kullanıldı?

Bu proje bir görüntü işleme projesi olsa da temel amaçlar; üzerinde çalışılan veriyi analiz edebilmek, analizlere göre veri üzerinde ekleme ,güncelleme, silme gibi hangi tür işlemlerin ne sıklıkla yapılacağına göre veriyi hangi veri yapısında saklamanın daha verimli olabileceğini analiz edebilmek ve veri üzerinde yapılacak işlemlere ve kendi risk kriterlerimize göre en verimli çalışacak algoritmaya karar verebilmektir. Bu kriterler projeden projeye değişebilir. Bellek kullanımı bizim için sıkıntı olmayabilir fakat çok hızlı bir şekilde işlemlerin gerçekleşmesini isteyebiliriz. Tam tersi bir durumda bizim için hızın önemi bellek kullanımından sonra gelebilir. Yukarıdaki işlemlerin yapılabilmesi için PGM formatı çok uygun bir format çünkü fotoğraf verisini okuyup yazabilmek bir metin dosyasına ulaşmak kadar basit ayrıca bu basitlik bizim fotoğraf verisine ulaşmaktan çok direkt olarak veriye ulaşıp yukarıda bahsedilen işlemlere odaklanmamıza imkan sağlıyor. Günümüzde RAW fotoğraflar profesyonel mecralarda sıklıkla kullanılıyor fakat bellekte kapladıkları alan çok yüksek olduğu için son kullanıcıya hitap eden alanlarda png , jpeg gibi dosya formatları daha çok tercih edilir çünkü bu formatlarda kamera sensörlerinden alınan bilgiyi direkt olarak kaydetmek yerine kameralardan girilen ayarlar parametre olarak alınarak (beyaz dengesi, Renk uzayı ,Bit derinliği, Dosya formatı) veri sıkıştırılır (Compress) .Bu sayede bellekten raw formatlara göre 3-4 kata kadar tasarruf edilir. Dolayısıyla bu dosya formatları ile çalışırken en başta belirtilen amaçlardan çok yeniden açma(Decompress) ve sıkıştırma işlemleri ile vakit kaybetmek istemiyoruz. Bu basitliğin anlaşılması için devam eden satırlarda PGM ve bir başka fotoğraf dosya formatı olan JPEG için görsel örnekler verilmiştir.

```
P2
# Shows the word "FEEP" (example from Netpbm man page on PGM)
24 7
15
0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
0  3  3  3  3  0  0  7  7  7  7  0  0 11 11 11 11  0  0 15 15 15 15  0
0  3  0  0  0  0  0  7  0  0  0  0  0 11  0  0  0  0  0 15  0  0 15  0
0  3  3  3  0  0  0  7  7  7  0  0  0 11 11 11  0  0  0 15 15 15 15  0
0  3  0  0  0  0  0  7  0  0  0  0  0 11  0  0  0  0  0 15  0  0  0  0
0  3  0  0  0  0  0  7  7  7  7  0  0 11 11 11 11  0  0 15  0  0  0  0
0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
```

https://en.wikipedia.org/wiki/Netpbm

https://en.wikipedia.org/wiki/Netpbm



https://sites.google.com/site/projectsummerinternship/home/basic-concepts-of-image-processing-and-jpeg-compression

Program Nasıl Derlenir ve Parametreler

Önce make edilmesi gerekiyor sonrasında komut satırına ./pgm_efekt (efektin ismi) (bulunduğu klasörle birlikte efekt uygulanacak resim) yazılmalıdır.Daha sonra .pgm.effectname biçiminde yeni fotoğraflar oluşur. Median işlemi için de komut satırına ./pgm_median yazılır ve sonuç olarak median_images klasöründe filtered.pgm resmi oluşur. Bu resim klasördeki diğer 9 adet resmin değerlendirilmesiyle diğer resimlerdeki turistin ortadan kaldırılmasıyla oluşan bir resimdir. Ayrıca program her çalıştığında komut satırında her resmin imzası,yorumu,en,boy ve kaç piksellik alan oluşturduğu yazar.

Valgrind Bellek Testi

Program kaynak dosyaları valgrind ile test edilmiştir. Alınan sonuçlar txt dosya formatında proje dosyaları arasına eklenmiştir.

Median Filtreleme

Bu projede median filtreme tekniğiyle 9 adet fotoğrafı girdi alarak fotoğraflarda bulunan turistin tamamen ortadan kaldırıldığı tek bir fotoğraf elde etmemiz isteniyor. Filtreleme operasyonunu nasıl yapacağımıza gelmeden önce median filtreleme yaklaşımı nedir bunu ele alalım.

Median filtreleme yöntemini görüntü işleme başlığının altında araştırdığınızda fark edilen şey, aslında çok yüksek oranda bu tekniğin fotoğrafçılıkta noise olarak nitelendirdiğimiz ve fotoğrafta istenmeyen noktalar halinde kendini belli eden görüntüyü ortadan kaldırmak için kullanıldığıdır. İstenmeyen bu parazitler çeşitli sebeplerle ortaya çıkabilir bunlar kamera sensörünün ışığa duyarlılığının az olması, Fotoğraf makinesinin ISO değerinin gereğinden fazla arttırılması, Sensör kalitesi gibi çeşitli nedenler olabilir. Bu metot lineer olarak çalışmayan bir metottur. Lineer olmamasının sebebi çalışma mantığı anlatıldığında daha açık olacaktır. Temel olarak fotoğraftaki her bir piksel için window olarak nitelendirilen 3*3 lük piksele denk gelecek şekilde bir inceleme işlemi yapılır. Her inceleme sonucunda bu alan bir piksel kaydırılır. Bu kaydırma

işlemini yaparken her bir pikselin tek bir satırda incelenerek sağındaki ve solundaki piksellere bakılması yerine matris yaklaşımıyla etrafındaki 8 piksele birden bakılır. Daha sonra her bir piksel değeri lineer olarak bir diziye aktarılıp sıralanır. Sonuç olarak bu dizide ortancaya denk gelen piksel incelenen pikselin değeri olarak seçilir. Burada noise denilen noktaların kaldırılabilmesinin sebebi bu noktaların fotoğrafta bulunan diğer piksellere göre çok siyah veya çok beyaz noktalar olmasıdır. Dolayısıyla bu pikseller, komşu pikselleri ile sıralandığında, ortancaya denk gelmeleri çok düşük bir ihtimaldir. Bu ihtimal fotoğrafta aşırı derecede noise olması durumlarında gerçekleşebilir. Böyle bir durumda ise 3*3 olan inceleme alanı 5*5 veya 7*7 olarak denenebilir. Fakat bu alanın artması durumunda ortaya çıkan fotoğraf noise'dan arınmasına rağmen netliğini daha fazla kaybeder.

Bu aşamadan itibaren median filtreleme konseptine artık daha fazla hakimiz. Bu projede uygulanan yaklaşım ise benzer fakat amaca ulaşılması için biraz daha farklı. Her bir pikselin komşu piksellerini bir diziye aktarıp sıralamak yerine aynı açıdan çekilmiş fakat turistin farklı konumlarda bulunduğu 9 farklı fotoğraftan, aynı konumda bulunan pikselleri karşılaştıracağız. Doğrusal olarak, farklı fotoğraflardan gelen piksellerin değerini sıraladıktan sonra ortanca olan pikseli filtrelenmiş fotoğrafı oluşturabilmek amacıyla alacağız ve yeni oluşturulacak fotoğraf dosyası için seçeceğiz. Ortancayı almamızın sebebi ise turistin hareket ettiğini ve 9 fotoğraftan bu uygulamanın başarılı olabilmesi için en azından 5 inde aynı konumda olmadığını biliyoruz. Bu durumda ortanca pikselin turistten değil peyzajdan geleceğini biliyoruz. Median filtrelemenin sakıncalı olabileceği noktalara alternatif çözümler konu başlığında ayrıca değinildi. Geçici dizideki piksel değerlerini sıralamak için bir sıralama algoritmasına ihtiyaç var bu algoritmanın seçimi ve median filtrelemenin teknik aytıntıları 3.bölümde ele alınmıştır.

Doğru Veri Yapısına Karar Vermek-Veri Yapısı Analizi

Elimizde bazı fotoğraf dosyaları var ve bu dosyalarda tutulan verileri okuyup üzerinde işlemler yapabilmek için bu verileri belli bir yapı ve düzen içerisinde bellekte tutmalıyız. PGM formatında çalışmak istediğimiz veriler fotoğrafın en ve boyutuna göre belli bir sayıdaki piksel verisini anlamına geliyor. Bizim durumumuzda her bir piksel için bu veri 0-255 arasındaki matematiksel bir tamsayı değeri olarak karşımıza çıkıyor. Bu noktada artık piksel verileri için int veri tipi ile çalışacağımızı da biliyoruz fakat programımızda ,veri yapısı olarak dizi, bağlı

listeler, Karma tablosu , yığın , kuyruk, gibi yapılardan hangisini seçmenin mantıklı olduğuna karar vermeliyiz. Bu kararı verebilmek için şu soruya cevap vermeliyiz : Bu veri ile ne yapmak istiyoruz. Bu projede yapmak istediğimiz işlemler ortanca filtreleme ve fotoğraflar üzerine bazı efektler uygulamak. Ortanca filtreleme işlemi için yapmamız gereken şey birden fazla fotoğraf dosyasından piksel verilerini okumak ,her bir fotoğrafta aynı sıraya denk gelen piksellerin değerlerini sıralamak ve ortanca olanı seçmek ve yazma işlemini gerçekleştirmek. Efekt işlemleri için ise; piksel değerlerini okumak, bu piksel değerlerinin yani 0-255 arasındaki matematiksel değerlerin üzerinde istenilen efektin gerekliliklerine göre matematiksel oynamalar yapmak ve yazma işlemlerini yapmak. Bu adımlara baktığımız zaman :

1) piksel verisine ulaşmanın en çok yapılan işlemlerden biri olduğunu

2) Çözünürlük değişikliği yapılmadığını daha açık ifade etmek gerekirse var olan piksellere yenilerinin eklenmesi veya çıkarılması durumunun olmadığı

3) Kümülatif bir veri ile çalışılmadığı

4)Her bir fotoğraf bazında piksel sayısının dosyadan alınan bilgilere göre net bir şekilde belli olduğu dolayısıyla ne kadar piksel verisiyle çalışılacağının fotoğraf dosyasını okunarak bilinmesi durumu

5)Piksellerin belli düzende dizilirlerse görsel olarak bir anlam ifade ettiği dolayısıyla 2 boyutlu olarak eğer x-y koordinat sisteminde düşünürsek, bir piksel onunla işlem yapılmadan önce nerede bulunuyorsa işlemden sonra da aynı konumda bulunmalı.

1.)Yukarıdaki maddeler göz önüne alındığında piksel değerlerinin tutulması için dizi veri yapısının bu senaryoya en uygun olanı olduğunu anlıyoruz bu veri yapısının en önemli avantajlarından birinin indeksler sayesinde diğer veri yapılarına göre veriye çok hızlı bir şekilde ulaşabilmek olduğunu biliyoruz

2)50x50 ebatlarında bir dosyayı örnek olarak alırsak bizim bu dosyayı 60x60 veya 40x40 gibi bir orana ,piksel ekleme veya çıkarma yaparak getirme niyetimiz yok. Eğer böyle bir amacımız olsaydı bu işlemin yapılma sıklığını ve işlemin bizim için maliyetini hesaba katmamız gerekirdi böyle bir durumda ekleme ve çıkarma işlemlerinde Bağlı listeler çok daha verimli olurdu. Dizide ise piksel ekleme ve

çıkarma durumlarında dizinin boyutunun değişecek olması bizim için verimsiz olurdu.

3)Sürekli ortaya çıkan ve veri tabanına eklenen bir veri yok .Örnek olarak bir kütüphane otomasyon sistemini ele alalım kullanıcı sürekli yeni kitaplar girebilir bu durumda istediğimiz kitapla ilgili bilgilere hızlıca ulaşmak isteyebiliriz bu durumda karma tablosu veya ilişkisel veri tabanı sistemlerini kullanmak isteyebiliriz fakat bu projede spesifik bir veriye hızlıca ulaşabilmek veya o veri için sorgu atmak gibi bir amacımız yok.

4)PGM dosyasındaki genişlik ve yükseklik bilgisini okumak bize ne kadar veri ile çalışacağımızın bilgisini veriyor. Dolayısıyla dizi veri yapısında çalışırken dizi boyutunu belirleme noktasında bizim için bir bilinmezlik dolayısıyla bir dezavantaj söz konusu değil.

5) Bir dizi kullanarak piksel değerlerini sırasıyla bu diziye aktarıp yine aynı sıra ile yazma işlemlerini gerçekleştirebiliriz bu durumda görsel olarak bir bozulma yaşamayız hem de diğer işlemlerimizde hızdan kaybetmeyiz. Örnek olarak piksel değerlerinin bir karma tablosunda tutmak ve her bir piksel değerini bucket'lara yerleştirmek ve piksele ulaşmak için pikselin matematiksel değeri üzerinden hash işlemi yapmak bizim durumumuzda olabilecek en verimsiz veri yapılarından biri olabilirdi.

Yukarıdaki 5 maddelik sebepler ve 5 maddelik analize göre projedeki veri yapısı olarak dizi (Array) seçilmiştir.

İskelet Koduna Ekleme Yapıldı Mı?

Proje gereksinimlerinde bizden seçeceğimiz bazı fotoğraf efektlerini projeye dahil etmemiz gerekiyordu. Bu yüzden efects.h başlık dosyasına ve efects.c kaynak dosyalarına sırasıyla fonksiyon tanımları ve fonksiyonların kendileri kodlanarak eklenmiştir. Bu fonksiyonlar parlaklık ve keskinlik fonksiyonlarıdır. Devam eden satırlarda sonradan eklenen bu fonksiyonlarla birlikte iskelet dosyasında gelen diğer efekt fonksiyonlarının çalışma mantığına kısaca değinilecektir. Bu metotların ürettiği sonuçlar ve zaman karmaşıklıkları 3. Bölümde ele alınacaktır.

Parlaklık Efekti

Bu projede siyah ve beyaz renkleri ve bu renkler arasındaki grinin tonlarını temsil etmek amacıyla her bir piksel için 0-255 arasındaki tam sayıları kullanıyoruz. 0 değeri siyah ,255 değeri ile ise beyaz rengi ifade ediyor. Arada kalan tam sayı değerleri ise siyah ve beyaz arasındaki tonları kapsıyor. Parlaklık efektinde yaptığımız şey ise her bir piksel için, belirlenen parlaklık faktörüne göre pikseli beyaz tonlara yaklaştırmak. Buradan anlıyoruz ki aslında fotoğraflarda düşük pozlama yaptığımız zaman da bu efektin tam tersi olarak siyah tonlara yaklaştırmış oluyoruz.

Terse Çevirme Efekti

Terse çevirme işlemi her bir pikselin değerini , o pikselin tanımlı olduğu aralıkta (0-255) tümleyen değeri ile değiştirmektir. Sonuç olarak Bir piksel eğer beyaz tona çok yakın ise artık siyah tona çok yakın olacak şekilde veya bir piksel açık gri ise o pikseli daha koyu tonda bir griye çevirecek şekilde fotoğraf üzerinde bir sonuç alırız.

Threshold Efekti

Yaptığımız işlem her bir piksel için parametre olarak alınan eşik değeri ile karşılaştırma yaparak bu değerin üstünde kalan değerleri beyaz altında kalan değerleri ise siyaha çevirir. Bu işlem basit bir if else yapısıyla kontrol edilir. Burada verilecek eşik değeri çok önemlidir. Eşik değeri beyaza çok yakın bir değer olarak verilirse neredeyse tamamen siyah bir fotoğrafla tam tersi durumda ise beyaza çok yakın bir değer olarak verilirse neredeyse beyaz bir fotoğrafla karşılaşabiliriz. Eşik değerinin ortanca değer olarak verilmesi durumunda ise fotoğrafın belirginliği açısından istisnalar dışında en en verimli sonucu alabileceğimizi söyleyebiliriz.

Yumuşaklık Efekti

Bu efektin temel mantığı pikseller arasındaki keskin geçişleri azaltmaya dayalıdır. Nesnelerin dokularını daha yumuşak bir hale getirir ve yüzeylerdeki

noktasal geçişleri tek bir noktada belirgin bir şekilde olmasından ziyade komşu pikselleri geçiş pikselleri gibi kullanır. Görsel olarak fotoğraf üzerinde bulur oluşturur

Proje Uygulanma Süresi - Yol Haritası – İletişim - Grup Çalışması

Projeyi uygulamak yaklaşık olarak 3 hafta sürdü. Görüntü işleme projesini yapmaya karar verdikten sonra grup olarak nasıl ilerleyeceğimizi düşünmek için çeşitli platformlar üzerinde buluştuk. Bu platformlar çoğunlukla WhatsApp ve Discord platformlarıydı. Bu platformlar üzerinden sesli ve görüntülü toplantılar yaptık. Öncelikle Projeyi iyi anlamamız gerekiyordu. Bu yüzden bizden neler istendiğini, projenin amacını, proje tanıtımından ve proje pdf'inden anladıklarımızı birbirimizle tartıştık. Bu aşamadan sonra çalışacağımız dosya formatını ve görüntü işlemedeki temel konseptleri daha iyi anlamamız gerekiyordu. PGM formatını ve çeşitli internet siteleri üzerinden görüntü işlemeye girişteki temel konseptler hakkında araştırmalarımızı yaptık ve tekrar görüştük. Bu noktadan sonra gereksinimler bizim için daha net bir hale geldi. Projeyi 3 parçaya böldük; birincisi fotoğraf dosyalarını okuma ve yazma fonksiyonları, ikincisi median filtrelemede kullanılacak sıralama algoritması ve diğer kodlar, üçüncüsü ise fotoğraflara uygulanacak resim efektleri oldu. Projeye başladığımızda henüz basit sıralama algoritmalarını öğrenmiştik fakat bu algoritmaların yavaş kalacağını derste öğrendiklerimiz ve araştırmalarımız sonucunda biliyorduk. Bu yüzden daha karmaşık olan ve divide and conquer yaklaşımıyla oluşturulmuş diğer algoritmaları araştırarak öğrendik. Bizim senaryomuza uygun en verimli algoritmayı ve veri yapısını seçmek için analiz yaptık. Yukarıdaki parçalara göre görev dağılımını tamamladıktan sonra istenilen fonksiyonları ve kodları her bölüm için ayrı ayrı tamamlayarak ilerledik. Tamamlanan her bölüm için tekrar görüştük ve karşılıklı olarak kodları inceledik. En son noktada test ve rapor yazma aşamalarını bitirdik.

Proje analizi ➔ Görüntü İşleme ve Format Analizi ➔ Projeyi Bölümlere Ayırma ve İş Bölümü➔ Algoritma ve Veri Yapısı Analizi ➔Toplantı ➔Okuma ve Yazma Fonksiyonları Yazımı➔Toplantı➔ Ortanca Filtreleme fonksiyonları➔ Toplantı ➔Eklenecek Efektlerin Araştırılması ➔Toplantı➔ Efekt Fonksiyonlarının Yazılması ➔ Test ➔ Rapor Hazırlama

Problemi Çözmek İçin Alternatif Yöntemler Denendi Mi?

Projede bizden istenilen şeyler çok netti. Bunlar median filtreleme ile turist kaldırmak ve bazı efektler uygulamak. Efektler konusunda yapılacak işlemler ve çözümler çok net. Bazı çözümler için çok alternatifimiz yok. Farklı bir dosya formatıyla çalışsaydık veya veriyi farklı veri yapılarında tutsaydık efekt fonksiyonları için çeşitli alternatifleri düşünmek zorundaydık. Median filtreleme ise projede direkt olarak bizden istenen bir yöntemdi fakat bu yöntemin çeşitli senaryolarda verimsiz kalabileceğini düşündük. Bu projede kullanılan turist içeren fotoğraflarda turistin kapladığı alan fotoğrafın kendisine oranla çok düşük. Eğer fotoğrafta nispeten daha büyük alan kaplayan bir cisim ve elimizde girdi olarak az fotoğraf olsaydı ve sonuç olarak bu cismi kaldırmak isteseydik kod üzerinde bazı değişiklikler yapmamızın gerekli olacağını fark ettik. Aksi halde seçilen piksellerin kaldırmak istediğimiz objeden gerekenden çok daha fazla gelmesi durumu ortaya çıkacaktı. Bu durumda kullanıcıdan bilgi almak ve geliştirilmiş farklı bir median filtreleme metodu yazarak kullanıcıya seçim yaptırılmasının uygun olabileceğini düşündük. Ayrıca median filtrelemenin tek bir fotoğraf üzerinde noise kaldırma işlemlerinde de verimli olabileceğini fark ettik. Fakat projede çalıştığımız fotoğraflar klasik median filtreleme yöntemiyle verimli bir şekilde sonuç veriyordu.

Alternatif yöntemleri denediğimiz yer sıralama algoritmaları kısmı oldu. Hız faktörünü görmek için farklı sıralama algoritmaları uyguladık ve programın kaç ms’de çalıştığını not aldık. Aktif olarak projede kullanılan algoritmanın en iyi sonucu vereceğini tahmin ediyorduk fakat somut olarak da gözlemiş ve emin olmuş olduk.

BÖLÜM 3

ALINAN ÇIKTILAR VE ANALİZLER

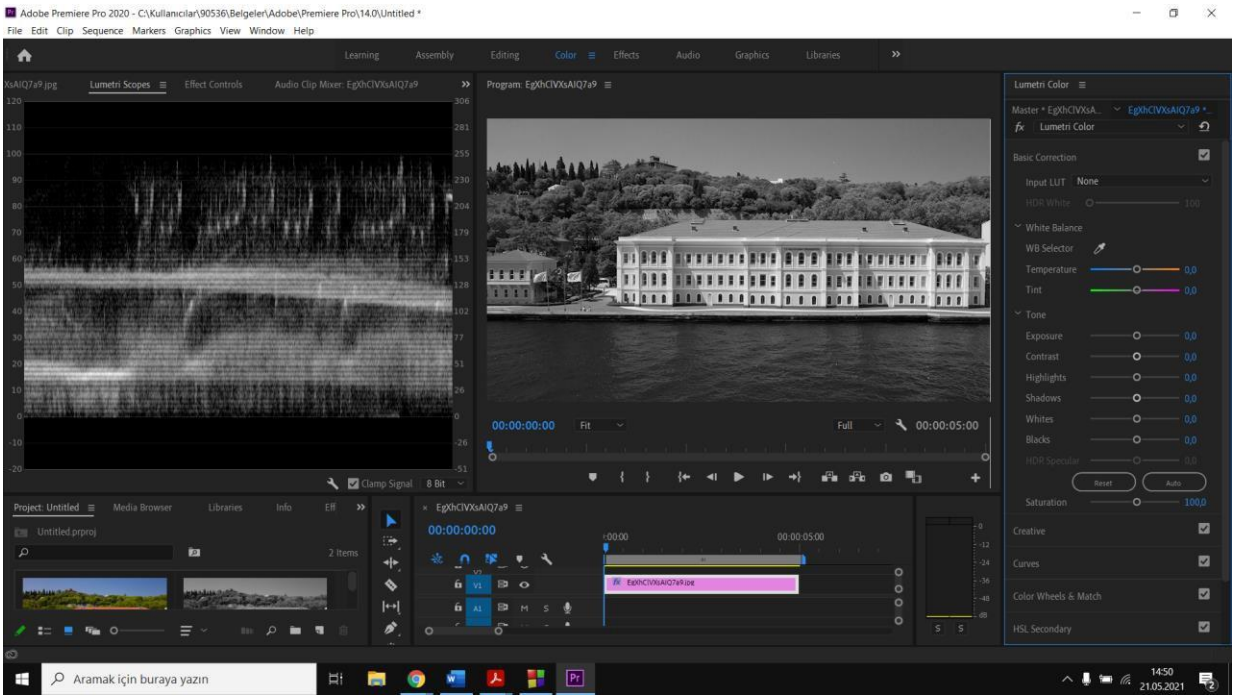
Programın, Metotların Verdiği Sonuçlar ve Fonksiyonların Zaman Karmaşıklığı Analizi

Program verilen fotoğraflardan turisti silerek yeni bir fotoğraf oluşturur. Ek olarak temel fotoğraf efektlerini de verilen fotoğraflara uygular. Dolayısıyla dosya okuma ve yazma işlemlerini de başarıyla gerçekleştirir. Devam eden satırlarda program metot bazında incelenecek, zaman karmaşıklığı ele alınacak ve görsel materyaller ile desteklenerek anlatılacaktır.

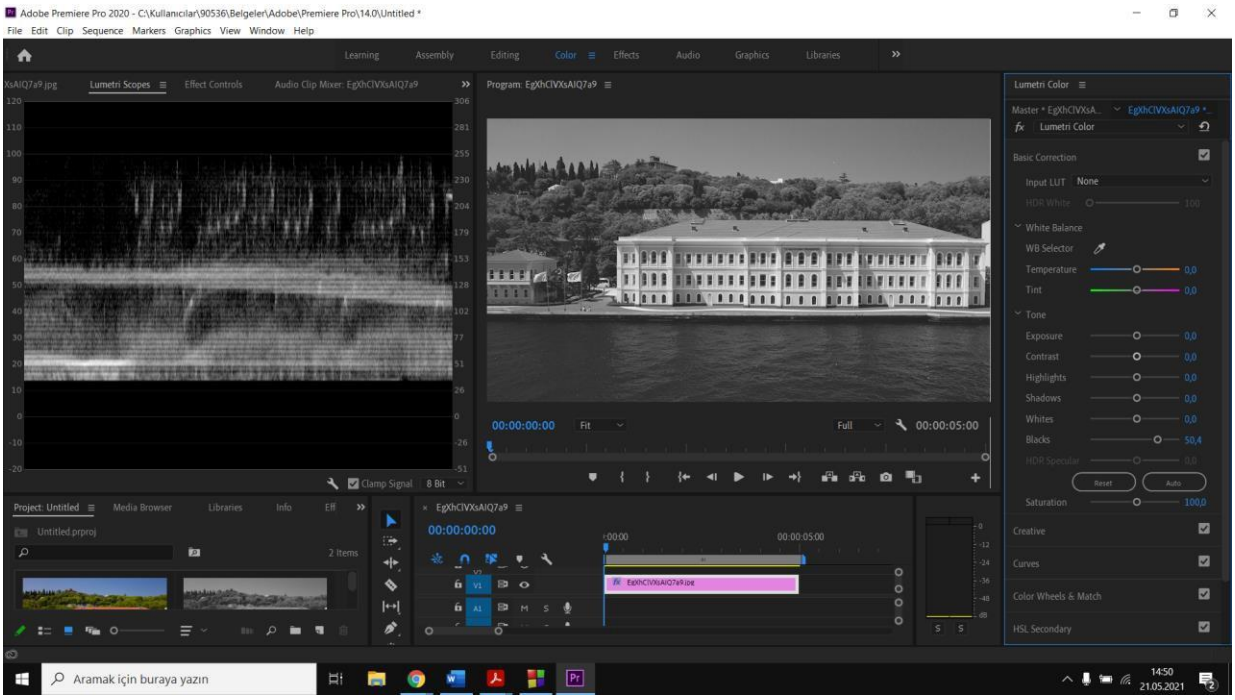
Parlaklık Efekti

Bu efektin çalışma mantığına proje mimarisinde değinmiştik. Dikkat edilmesi gereken noktalardan ilki ne kadar parlaklık efekti uygulanmak istendiğine karar vermek, 2. Önemli nokta ise bu parlaklık faktörünün her bir piksele ve eşit bir şekilde uygulanması. Bu noktada eşit bir şekilde her bir piksele uygulamak demek fotoğrafın tamamında pikselin beyaz tona ne derece yakın veya siyah tonlara ne derecede yakın olduğunu önemsemiyoruz demek. Eğer piksellerin matematiksel değerleri ile doğru veya ters orantılı olarak bu piksel değerlerine ekleme yaparsak ,bu efekte artık parlaklık efekti diyemeyiz. Yapılan işleme göre bu efektler siyah tonları daha siyaha çekebilen veya beyaz tonlara daha çok etki eden efektlerden olan Highlight ,Shadows , Whites, Black ayarlarına girer .Bu açıklamanın daha anlaşılır olması için devam eden satırlarda waveform grafikleri ile görsel örnekler verilmiştir.

Grafik gösterimi için programda kullanılan GSÜ fotoğrafı, GSÜ resmi twitter heabından alınmıştır.



Yukarıdaki fotoğrafta sol tarafta bulunan waveform grafiği bize piksellerin değerleri ve bulunduğu alanlar hakkında bilgi veriyor. Yukarıdan aşağıya doğru sıralanmış değerler, siyah ve beyaz arasındaki değerleri matematiksel olarak ifade ediyor. Aşağıdaki görüntüye baktığımız zaman ise mavi kutucuk içine alınmış alanda Black efektinin pozitif olarak uygulandıktan sonra sol tarafta waveform grafiğinde meydana gelen değişiklikleri görüyoruz grafikten anlaşıldığı üzere siyah tonları yukarı çekerken beyaz tonlara neredeyse etki etmedi yukarıda bahsedilen ayrım tam olarak budur.



Parlaklık efektinin ne olduğunu detaylıca açıkladık ve benzer efektlerle karıştırılmaması için örnekler verdik şimdi ise fonksiyondaki kodu ,yapılan açıklamalar doğrultusunda hızlıca açıklayalım. En boy bilgisini ve pikselleri saklayan dizinin başlangıç adresini parametre olarak geçtik. Döngü içerisinde her bir pikselin değerine 25 tam sayısını ekledik ve cur değişkenine atadık .Eğer cur(piksel) değeri toplama işleminden sonra 255'i geçiyor ise(cur) 255 değerini o piksele değer olarak verdik .Buradaki mantık şu ;bir pikselin alabileceği maksimum değerin 255 yani beyaz rengin kendisi olduğu için 255'in üstüne çıkamayız. Dolayısıyla 255 i aşıyorsa zaten beyaz renge ulaşmışız demektir. If else bloğununun gerekliliği bu yüzdendir.

Bu metodun zaman karmaşıklığını N olarak ifade etmeliyiz. N işlem yapmamız gereken piksel sayısını ifade eder. N olmasının sebebi istisnasız bütün piksellerin değerini değiştirmemiz gerektiğidir bu yüzden for döngüsü ile bütün pikselleri gezmemiz şart. Dolayısıyla en iyi ve en kötü durumda zaman karmaşıklığı yine N olacaktır. Bu metodun hızı diğer faktörleri sabit alırsak, fotoğrafın içerdiği piksel boyutuna göre bağıldır. Devam eden satırlarda bu metot ile ilgili görsellere ve grafiklere yer verilmiştir. Parlaklık faktörü 25 olarak seçilmiştir fakat farklı değerler de seçilebilir.

https://www.allaboutcircuits.com/technical-articles/digital-image-processing-point-operations/

Yukarıdaki görseller 70 faktöründe uygulanan parlaklık efektinin nasıl göründüğünü ve piksel bazında grafik olarak ifadesini gösterir. Yukarıdaki grafik ,veriyi matris olarak depoladığımız senaryoyu içerir fakat grafik olarak aynı görüntüyü verir.

Terse Çevirme Efekti

Bu Efektin Çalışma mantığına proje mimarisi bölümünde değinmiştik. Burada ise aşağıda bulunan 2 örnek soldaki orijinal fotoğrafa terse çevirme işlemi uygulandıktan sonra görsel olarak nasıl bir çıktı aldığımızı gösteriyor. Görselleri incelediğimizde kadının saçlarının çok koyu tonlarda olduğunu fark edebiliyoruz. Tersine çevirme efektini uyguladıktan sonra ise koyu olan saçların beyaza çok yakın tonlara dönüştüğünü görüyoruz. Diğer bir örnek olarak soldaki fotoğrafta sağ altta gözüken beyaz nesneye baktığımızda terse çevirme işleminden sonra sağdaki fotoğrafta nesnenin çok koyu bir hale geldiğini görüyoruz. Sonuç olarak efekt çok koyu ve çok açık alanlarda keskin bir değişiklik yaparken orta tonlarda görsel olarak daha az değişiklik meydana geliyor.

Bu metodun zaman karmaşıklığını N olarak ifade edebiliriz. N işlem yapmamız gereken piksel sayısını ifade eder. N olmasının sebebi istisnasız bütün piksellerin değerini değiştirmemiz veya üzerinde matematiksel işlem yapmamız gerekliliğinden dolayı for döngüsü ile her pikseli gezmemiz gerekliliğidir. Bu yüzden en iyi ve en kötü durum zaman karmaşıklığı analizimiz yine N'dir .Diğer etmenleri sabit tutarsak bu metodun hızı yine fotoğrafın içerdiği piksel sayısına göre bağıldır.

Bu metodu kodlarken en, boy ve pointer parametre olarak alarak her bir pikselin değerini for döngüsü kullanarak 255 den çıkarıyoruz ve yeni değeri olarak atıyoruz. 255 değerinden çıkarmamızın sebebi projemizdeki fotoğraflarda piksellerin değer aralığının 0 ve 255 aralığında olmasıdır.

Siyah ve Beyaz Efekti

Bu metot diğerlerinde olduğu gibi pointer ,yükseklik ve genişlik değerlerinin yanında referans olarak kullanılacak eşik değerini de parametre olarak alır. Eşik değerini for döngüsü içerisinde gezerek her bir pikselin değeri ile karşılaştırır. Büyüklük ve Küçüklük durumuna göre hangi koşula uyuyor ise o pikselin değerini 0 ve 255 ten birini seçerek değiştirir. Aşağıda bu efektin ürettiği sonuçları görsel olarak verilmiştir.

Bu metodun zaman karmaşıklığını N olarak ifade edebiliriz. N işlem yapmamız gereken piksel sayısını ifade eder. N olmasının sebebi istisnasız bütün piksellerin değerini değiştirmemiz gerekliliği ,değiştirmesek bile kontrol etme gerekliliğinin olmasıdır. Bu yüzden en iyi ve en kötü durumlarda da zaman karmaşıklığı yine N'dir .Bu metodun hızı fotoğrafın içerdiği piksel sayısına göre bağıldır.

Yumuşaklık Efekti

Bu efektin çalışma Mantığına proje mimarisinde değinmiştik .Kod olarak incelediğimizde yine pointer, en ve boy parametrelerini alır ve sırasıyla dizi üzerinde for döngüsü kullanarak ilerler ve mevcut pikselin değerini o pikselin o anki değeri ve bir sonraki pikselin değerinin ortalamasını alarak değiştirir. Böylece pikseller arasındaki keskin ton geçişlerini komşu piksellerle ilişkisine bakarak azaltır.

Bu metodun zaman karmaşıklığını N olarak ifade edebiliriz. N fotoğrafta bulunan piksel sayısını ifade eder. N olmasının sebebi for döngüsü ile istisnasız bütün piksellerin değerini değiştirmemiz veya üzerinde matematiksel işlem yapmamız gerektiğidir. Bu yüzden en iyi ve en kötü durum analizimize göre yine N değerine ulaşıyoruz. Bu metodun hızı diğer faktörleri sabit tuttuğumuzda fotoğrafın içerdiği piksel sayısına göre bağıldır.

Keskinlik Efekti

Bu metodun zaman karmaşıklığı N'dir parametre olarak pikselleri barındıran dizi, yükseklik ve genişlik değerlerini alır. For döngüsü kullanılarak baştan başlayıp son piksele kadar piksellerin değerine bakılır eğer piksel değerinin 2 katı 255 değerini geçiyor ise, max değerimiz 255 olduğu için piksele değer olarak 255 değeri verilir. Piksel değeri 2 ile çarpma işleminden sonra 255'i geçmiyorsa değer olarak çarpma işleminden sonraki değeri yeni değeri olarak atanır.

Median Filtreleme ve Algoritma Analizi

Bu filtrelemenin gerek bu projede gerekse farklı amaçlar için nasıl kullanıldığına ve çalışma mantığına 2.Bölümde değinmiştik. Ortanca değerin seçilebilmesi için farklı fotoğraflardan gelen piksel değerlerinin sıralanmasına ihtiyacımız var. Bu sıralama işlemi için bir algoritma seçimi yapılması gerekiyor. Projenin ilk aşamalarında basit sıralama algoritmalarından olan insertion sort algoritması, hem sonucu görebilmek hem de performans farkını gözlemleyebilmek adına seçilerek kodlandı. Fakat daha hızlı olan, parçala ve yönet konseptinde tasarlanmış algoritmalardan birini analiz ederek projeye uygulamamız performansı arttırmak için gerekliydi. Bu aşamada en iyi algoritmayı seçebilmek için çalıştığımız veriyi ek olarak gelişmiş sıralama algoritmaları olan merge sort ve quick sortu incelmeye başladık. Merge sort ve Quick Sortu ele aldığımızda quicksort algoritmasının ekstra bellek kulanımı yapmadan varolan dizi üzerinde sıralama yapması, mergesort algoritmasına göre bir avantaj. Fakat işlenmesi gereken veri incelendiğinde projede çalışılan fotoğrafların çözünürlüklerinin düşük olması ayrıca çok sınırlı sayıda fotoğraf üzerine işlem yapıldığı gerçeği, merge sortun daha hızlı çalışması durumunda, mergesort algoritmasının ekstra bellek kulanımı bizim için göz ardı edilebilir bir durumdu. Bu noktadan sonra bellek kullanımı faktörü bizim için nötr bir faktör haline geldi ve tamamen hıza odaklandık. Hız konusuna geldiğimizde ise sıralanması gereken veriyi göz önüne aldık. Bu veri, bir dizi veri yapısında ve 9 farklı fotoğraftan gelen 9 farklı piksel değeri anlamına geliyor. Her bir aşamada 9 değerli bir diziyi sıralamamız gerektiği göz önüne alındığında ,bu veri kümesinin küçük bir veri kümesi olduğu açık bir şekilde belli. Her ne kadar en kötü senaryoda quicksort'un O(N2) ve mergesort'un O(nlogn) olsa da . Pratikte QuickSort algoritmasının küçük veri kümelerinde ve dizi veri yapısında merge sort algoritmasına göre daha performanslı çalıştığını biliyoruz. Bu veri yapısı bağlı listeler olsa idi performans testi yaparak, merge sort tekrar gözden geçirilebilirdi. Fakat bu koşullarda, quicksort'un avantajlı olduğu durumlar projemizdeki durumlarla çakışıyor ve bu algoritmayı bizim projemizde zaman karmaşıklığı açısından avantajlı hale getiriyor. Bu sebeplerden dolayı quicksort algoritması bu projede seçilen sıralama algoritması olmuştur. Sonuç olarak bu metot turistin ortadan kalktığı fotoğrafı başarıyla oluşturur.

BÖLÜM 4

ÖZET VE GELECEK UYGULAMALAR

Sonuçlar

Projede bizden istenilen uygulamaları Gerçekleştirdik. Fotoğraf verilerini okuma ve yazma işlemlerini yaptık. Verilen fotoğraflardan turisti kaldırabildik. Çeşitli fotoğraflara temel fotoğraf efektlerini uygulayarak efektli yeni fotoğraflar oluşturma işlemlerini tamamladık.

Kazanımlar

Projeyi tamamladıktan sonra karşılaşılan veriye göre kullanacağımız veri yapılarını nasıl seçmeliyiz, sorgulamamız gereken kriterler nelerdir, senaryolara göre en verimli olabilecek algoritmaya nasıl karar verebiliriz, Program üzerinde hangi işlemler daha fazla yapılacak ve bu işlemlerin maliyetlerinin analizi gibi konularda deneyim kazandık. Ek olarak görüntü işlemeye girişteki konseptlerin neler olduğunu öğrendik. Görüntü işlemenin kullanıldığı farklı sektörleri ve bu sektörlere getirilen farklı çözümleri ve yaklaşımları inceledik.

Projenin Geleceği

Projeye daha fazla fotoğraf efekti ekleyerek ve programın ara yüzünü çeşitli araçlarla konsol ekranından daha kullanıcı dostu bir formata çevirebiliriz. Sonuçta bir program en iyi çözümleri sunsa ve en verimli kaynak yönetimine sahip olsa bile kullanıcı ile iletişime geçemiyorsa bu büyük bir problemdir. Eklenebilecek özelliklerde biri de uygulanan fotoğraf efektlerinin hangi oranlarda uygulanacağını kullanıcıdan alarak işlem yapmak olabilir. Bu işlem programı daha kullanışlı hale getirerek bir adım ileriye taşıyabilir.

REFERANSLAR

https://en.wikipedia.org/wiki/Netpbm

https://www.unrealengine.com/en-US/industry/film-television

https://www.youtube.com/watch?v=QlVK96MH-vE

https://sites.google.com/site/projectsummerinternship/home/basic-concepts-of-image-processing-and-jpeg-compression

https://www.cs.auckland.ac.nz/courses/compsci373s1c/PatricesLectures/Image%20Filtering_2up.pdf