TARANMIŞ GÖRÜNTÜLERDE EĞİK BELGE BULMA VE DÜZELTME

Mücahit Durmaz
181419203

MEZUNİYET PROJESİ

Bilgisayar Mühendisliği Anabilim Dalı
Bilgisayar Mühendisliği Tezsiz Yüksek Lisans Programı
Danışman: Dr. Öğr. Üyesi Erdal Güvenoğlu

İstanbul
T.C. Maltepe Üniversitesi
Lisansüstü Eğitim Enstitüsü
Haziran, 2020

TARANMIŞ GÖRÜNTÜLERDE EĞİK BELGE BULMA VE DÜZELTME

Mücahit Durmaz
181419203
Orcid: 0000-0003-4066-3004

MEZUNİYET PROJESİ
Bilgisayar Mühendisliği Anabilim Dalı
Bilgisayar Mühendisliği Tezsiz Yüksek Lisans Programı
Danışman: Dr. Öğr. Üyesi Erdal Güvenoğlu

İstanbul
T.C. Maltepe Üniversitesi
Lisansüstü Eğitim Enstitüsü
Haziran, 2020

TEŞEKKÜR

Yüksek lisans eğitimim süresince ve bu tezin hazırlanmasında bana yol gösteren, bilgi, tecrübe ve desteklerini esirgemeyen saygıdeğer danışman hocam Dr. Öğr. Üyesi Erdal Güvenoğlu' na ve destekleriyle bugünlere ulaşmamı sağlayan aileme sonsuz teşekkürlerimi sunarım.

Mücahit Durmaz

Haziran 2020

ÖZ

TARANMIŞ BELGE GÖRÜNTÜLERİNDE EĞİKLİK BULMA VE DÜZELTME

Mücahit Durmaz
Mezuniyet Projesi
Bilgisayar Mühendisliği Anabilim Dalı
Bilgisayar Mühendisliği Yüksek Lisans Programı
Danışman: Dr. Öğr. Üyesi Erdal Güvenoğlu
Maltepe Üniversitesi Lisansüstü Eğitim Enstitüsü, 2020

İnsanların iş yapış şekilleri günden güne dijitalleşmeye başlamıştır. Bir belgeyi fiziksel olarak saklamak hem yer kaplamakta hemde kolay ulaşımını zorlaştırmaktadır. Bu yüzden belgelerin dijital ortama taşınması için belgeler cihazlarda taratılır veya fotoğrafı çekilerek dijital ortama aktarılır. Taranmış belge veya fotoğrafı çekilmiş görüntülerde eğiklik olması kaçınılmazdır.

Bu çalışmada taranmış veya fotoğrafı çekilmiş belge görüntüsündeki eğiklik bulunarak düz bir görüntü elde etmek amaçlanmıştır. Sisteme gelen görüntüye çeşitli görüntü işleme teknikleri uygulanır. Görüntü üzerinde daha kolay işlemler uygulayabilmek için görüntü önce gri görüntüye dönüştürülür. Sonrasında gri görüntü üzerinde filtrelemeler ile iyileştirmeler yapılarak görüntüde kenarların bulunması kolaylaştırılır. Görüntüde kenarlar bulunduktan sonra görüntü içindeki belgenin kenar noktaları belirlenerek geometrik görüntü dönüşümleri uygulanır. Bu yapılan işlemler sonucunda görüntü içerisindeki eğik belge, düz bir biçimde olmak üzere çıktı olarak üretilir.

Anahtar Sözcükler: Geometrik Görüntü Dönüşümleri, Kenar Bulma, Belge Eğriliğini Düzeltme.

ABSTRACT

FINDING AND CORRECTING SKEW IN SCANNED DOCUMENT IMAGES

Mücahit Durmaz

Graduation Project

Department of Computer Engineering

Computer Engineering Without Thesis Programme

Thesis Advisor: Asst. Prof. Erdal Güvenoğlu

Maltepe University Graduate School, 2020

The way people do business has begun to digitalize day by day. Storing a document physically both takes up space and makes it difficult to access. For this reason, documents are scanned on the devices or transferred to digital media to take the documents to digital media. It is inevitable to have a skew in the scanned document or photographs taken.

In this study, it is aimed to obtain a flat image by finding a skew in the scanned or photographed document image. Various image processing techniques are applied to the image that comes to the system. The image is first converted to a gray image to make it easier to perform operations on the image. Afterwards, improvements are made by filtering on the gray image, and the edges are easier to find in the image. After the edges are found in the image, the edge points of the document in the image are determined and geometric image transforms are applied. As a result of these operations, the oblique document in the image is produced as an output, in a flat form.

Keywords: Geometric Image Transforms, Edge Finding, Correcting Document Curvature.

İÇİNDEKİLER

ŞEKİLLER LİSTESİ

ÖZGEÇMİŞ

Mücahit Durmaz

Bilgisayar Mühendisliği Anabilim Dalı

Eğitim

Derece	*Yıl*	*Üniversite, Enstitü, Anabilim/Anasanat Dalı*
Ls.	2017	Karadeniz Teknik Üniversitesi, Mühendislik Fakültesi Bilgisayar Mühendisliği Anabilim Dalı
Lise	2012	Hasan Türek Anadolu Lisesi

İş/İstihdam

Yıl	*Görev*
2018 -	IT Engineer, Turkcell İletişim Hizmetleri
2018 - 2018	Yazılım Mühendisi, TRSIM-Robos Endüstriyel Otomasyon Şti.

Kişisel Bilgiler

Doğum yeri ve yılı : Manisa, 1994 Cinsiyet: E

Yabancı diller : İngilizce (Orta)

e-posta : mucahit_durmaz@hotmail.com

BÖLÜM 1. GİRİŞ

Son yıllarda teknolojik gelişmelerle dokümanları dijital ortama taşımak gayet kolaylaşmaya başlamıştır. Örneğin eski ve yıpranmaya yüz tutmuş Osmanlı ve Cumhuriyet dönemi arşiv dosyalarının teknolojinin yardımıyla dijital ortama aktarılma çalışmaları başlamıştır. Bu çalışmalarla hem dosyaların yıpranarak eskimesinin engellenmesi hem de kolay erişilebilir olması amaçlanmaktadır. Bir diğer örnek olarak herhangi bir bankanın müşterisi olmak veya kredi çekmek için onlarca belge imzalanmaktadır. Bu belgeler imzalandıktan sonra tarayıcı veya fotoğrafı çekilerek dijital ortama aktarılmaktadır. Böylelikle belgelerin kolay ulaşılabilir olması sağlanmaktadır. Öte yandan bu tarama veya fotoğraf çekilerek yapılan belgenin dijitalleştirilmesi işleminde görüntü içerisindeki belgenin eğik olması kaçınılmazdır.

Bu çalışma ile taranmış veya fotoğrafı çekilmiş belgede sistem tarafından görüntü içerisindeki belgenin bulunup eğikliğinin giderilmesi üzerinde durulmuştur. Böylelikle görüntü içerisindeki gereksiz detaylar elimine edilip ana istenen bilgiye direkt olarak ulaşılmış olunur. Bu işlemlerin nasıl yapıldığı Literatür Araştırması bölümünde şu başlıklar altında incelenmiştir.

- Gri Tonlama
- Bilateral Filtreleme
- Adaptive Threshold
- Median Blur
- Canny Kenar Belirleme
- Contour
- Perspective Dönüşüm

BÖLÜM 2. LİTERATÜR ARAŞTIRMASI

Bu bölümde taranmış veya fotoğrafı çekilmiş bir görüntünün içerisindeki eğik belgenin üzerinde çeşitli algoritmalar ve yöntemler uygulanarak nasıl düz bir forma dönüştürüleceği ile literatür araştırmaları anlatılmıştır. Görüntü üzerinde işlem yapılabilmesi griye dönüşüm ile binary dönüşümler , bulanıklaştırma yöntemleri, kenar bulma algoritmaları, konturlama ve dönüşüm algoritmaları incelenmiştir.

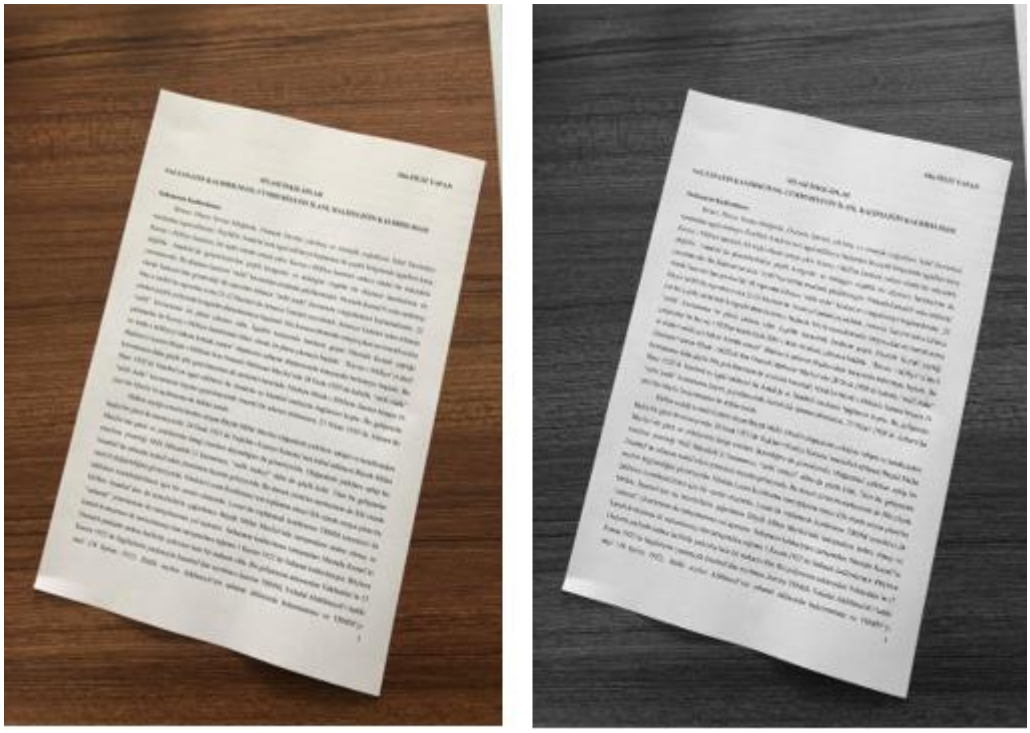
2.1. Gri Tonlama Dönüşümü

Gri Tonlama, görünür rengi olmayan gri tonları serisidir. Olası en koyu gölge, iletilen veya yansıyan ışığın tamamen yokluğu olan siyahtır. Mümkün olan en açık gölge beyazdır. Ara gri tonları, iletilen ışık için üç ana rengin (kırmızı, yeşil ve mavi) eşit parlaklık miktarlarıyla temsil edilmektedir.

Gri seviyeli bir görüntüde her piksel 0 (siyah) ‘dan 255(beyaz)’a kadar farklı tonlamaya sahiptir. Gri seviyeli bir resimde grinin tonları olduğu için siyah beyaz resim ile karıştırmamak gerekmektedir. Çünkü siyah beyaz resimde sadece siyah ve beyaz renkleri bulunmaktadır. İletilen ışık durumunda (örneğin, bir bilgisayar ekranındaki görüntü), kırmızı (R), yeşil (G) ve mavi (B) bileşenlerin parlaklık seviyelerinin her biri 0 ila 255 arasındaki bir sayı olarak temsil edilmektedir. Grinin açıklığı, RGB değerleri birbirine eşit olduğu için parlaklık düzeylerini temsil eden sayıyla doğru orantılıdır [1].

Görüntünün gri seviyeye dönüştürülmesinin başlıca nedenleri şunlardır:

- Gürültüyü engellemesi
- Kod karmaşıklığının azaltılması
- Daha hızlı işlem yapılabilmesi



Şekil 1. Görüntünün Gri Görüntüye Çevirilmesi

2.2. Görüntü Yumuşatma

Görüntüleri yumuşatma (Smoothing Images), gürültüyü gidermek için kullanışlıdır. Görüntüden yüksek frekanslı içeriği (Örn. Parazit, Kenarlar) filtre matrisiyle kaldırarak görüntüyü kalitelileştirmektedir. Böylelikle gürültü giderilmiş kenarlarda bulanıklaştırılmış olmaktadır.

Filtreleme, görüntü işleme ve bilgisayarla görmenin belki de en temel işlemidir. Filtreleme, belirli bir konumdaki filtrelenmiş görüntünün değeri aynı konumdaki komşu giriş görüntüsünün değerlerinin bir fonksiyonudur.

2.2.1. Ortalama Bulanıklaştırma Filtrelemesi

Görüntü üzerinde belirlenmiş kutu filtre matrisi gezdirilerek, filtrenin denk geldiği alandaki piksel değerlerinin ortalaması alınıp kutu filtre matris merkezinin denk geldiği piksele bu değer yazılır. Bu işlem tüm pikseller için tek tek uygulandıktan sonra bulanık görüntü elde edilmiş olmaktadır [2].

2.2.2. Gaussian Methoduyla Bulanıklaştırma Filtrelemesi

Bu yöntemde gaussian filtre matrisi kullanılarak bulanıklaştırılmış görüntü elde edilmektedir. Gauss bulanıklığı, görütüden Gauss gürültüsünün giderilmesinde oldukça etkilidir [2].

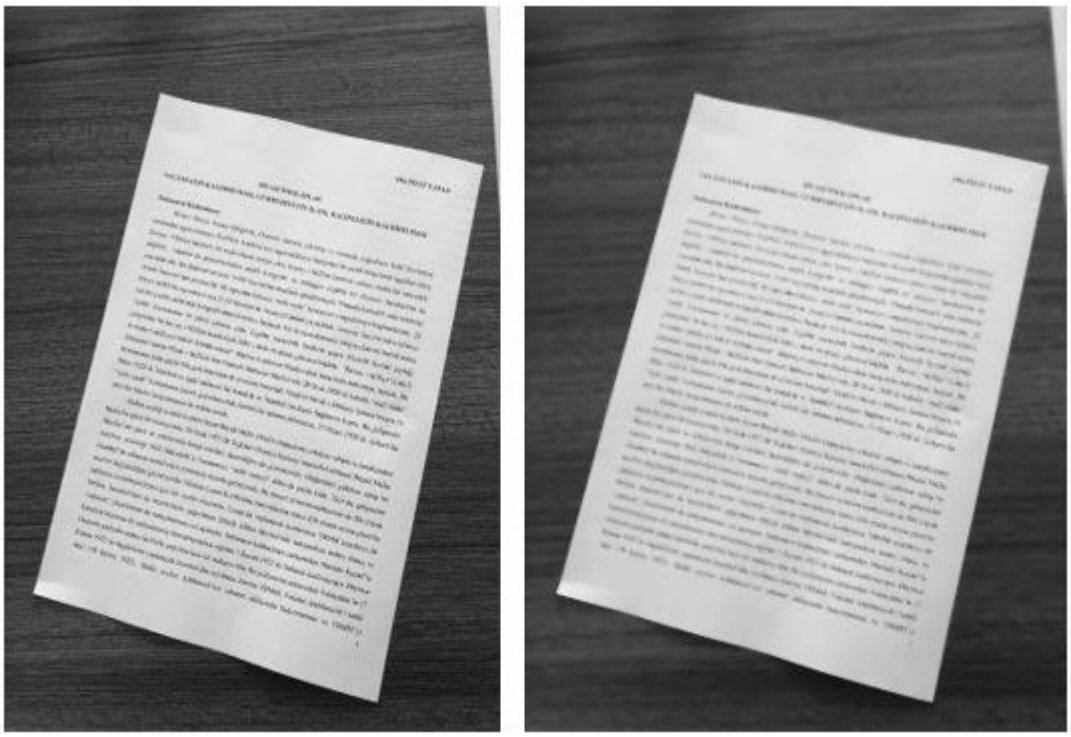
2.2.3. Medyan Bulanıklaştırma Filtrelemesi

Bu yöntemde filtreleme matrisi tüm piksellerde gezdirilerek, filtreleme matrisinin denk geldiği alanın medyanı hesaplanıp filtre matrisinin denk geldiği merkez piksele yazılır bu şekilde gürültü etkili bir şekilde azaltılmaktadır [2].

2.2.4. İki taraflı Bulanıklaştırma Filtrelemesi

İki taraflı (Bilateral) bulanıklaştırma filtresi kenarları keskin tutarken gürültü azaltmada etkili bir yöntemdir. Ancak işlem diğer yöntemlere göre daha yavaştır.

İki taraflı filtreleme de uzayda bir Gauss filtresi alır, ancak piksel farkının bir fonksiyonu olan bir Gauss filtresi daha almaktadır. Gauss işlevi, yalnızca yakındaki piksellerin bulanıklaştırma için dikkate alınmasını sağlarken, Gaussian yoğunluk farkı işlevi, yalnızca merkezi piksele benzer yoğunluklara sahip piksellerin bulanıklaştırma için dikkate alınmasını sağlamaktadır. Böylece kenarlardaki pikseller büyük yoğunluk varyasyonuna sahip olacağından kenarlar korunmaktadır [2].



Şekil 2. İki Taraflı Eşikleme Uygulanmış Görüntü

2.3. Görüntü Eşikleme

Görüntünün segmentlere ayrılmasında en basit yöntem görüntü eşiklemedir. Görüntüdeki her piksel için threshold değeri sorgulanarak görüntü gri görüntüden siyah beyaz görüntüye çevirilir [3].

2.3.1. Basit Eşikleme

Her bir piksel için önceden belirlenmiş olan aynı eşik değeri kullanılarak, piksel değeri eşik değerden küçükse 0 (siyah) büyükse maximum değer olan 255(beyaz) olarak setlenmektedir. Bu şekilde her piksel için threshold karşılaştırılması yapılarak görüntü binary biçime yani siyah ve beyaz biçimine dönüştürülerek yeni işlemlere hazır hale getirilmektedir.

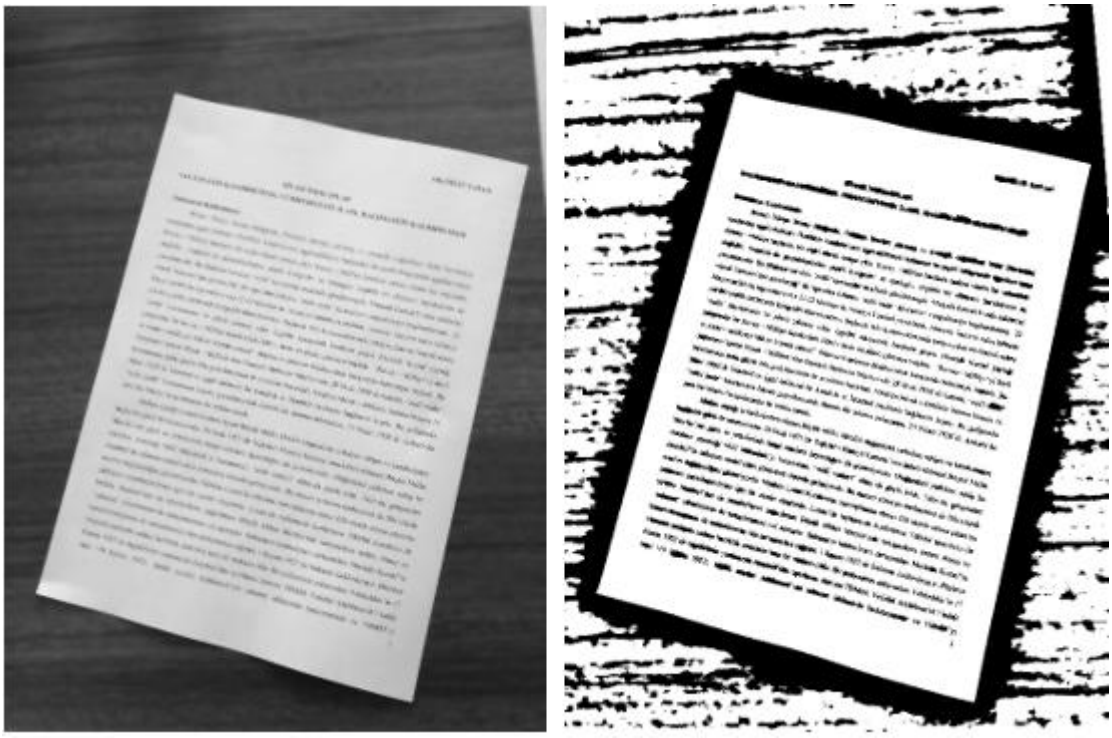
2.3.2. Adaptif Eşikleme

Basit eşikleme metodunda önceden belirlenmiş threshold değerine göre işlemler yapılmaktaydı. Ancak bu metot her koşulda iyi sonuç vermeyebilmektedir. Bunun önüne

geçmek için giriş resmine bağımlı olan Adaptif Eşikleme metotu kullanmak daha doğru olacaktır. Burada algoritma etrafındaki komşu piksel değerlerine göre eşik belirler. Böylelikle aynı görüntünün farklı bölgeleri için farklı ışık eşik değerleri elde ederek farklı aydınlatmaya sahip görüntüler için daha iyi sonuçlar vermektedir.

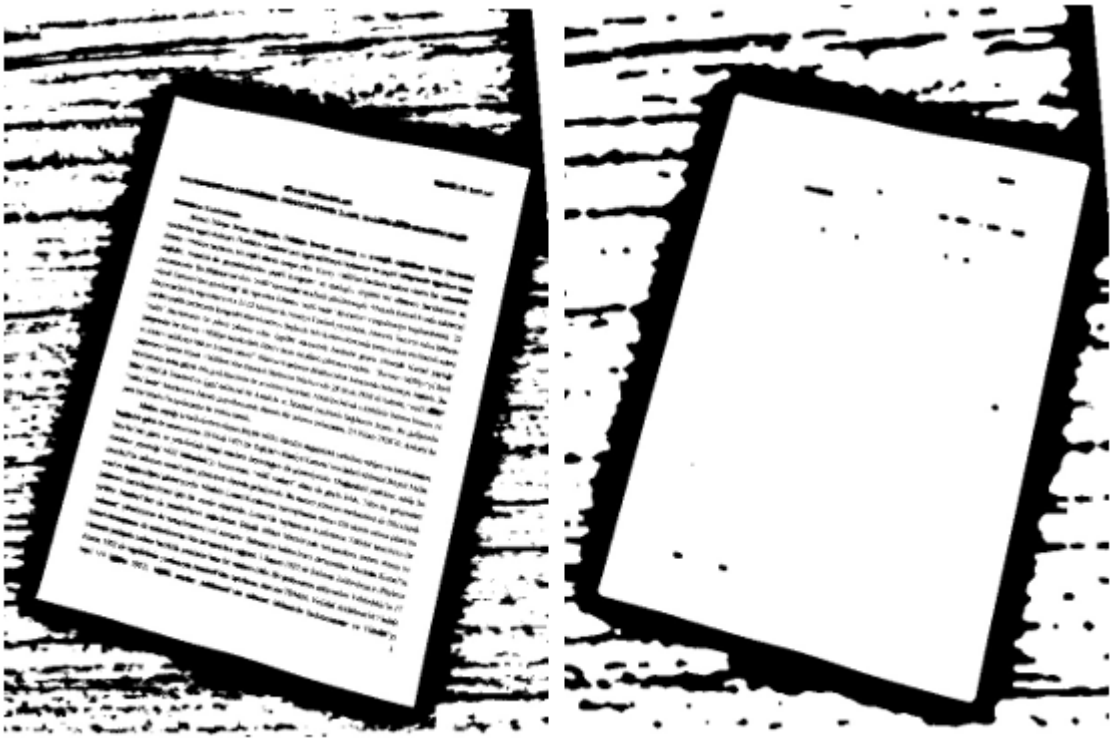
2.3.3. Otsu Eşikleme

Otsu Eşikleme değeri görüntünün histogram ortalamasını eşik değeri olarak almaktadır. Örnek olarak histogramın yalnızca iki pikten oluşacağı, yalnızca iki ayrı görüntü değerine (bimodal görüntü) sahip bir görüntüde, eşik değer bu iki değerin ortasında iyi bir eşik olacaktır. Benzer şekilde Otsu yöntemi, görüntü histogramından en uygun global eşik değerini belirlemektedir.



Şekil 3. Adaptif Eşikleme Uygulanmış Görüntü

Çalışmada Adaptif uygulanmış görüntüye medyan bulanıklaştırma filtrelemesi uygulanarak küçük detaylar elimine edilmiştir.



Şekil 4. Medyan Bulanıklaştırma Uygulanmış Görüntü

2.4. Kenar Belirleme Algoritmaları

Kenar belirleme, görüntü işlemenin en temel konularından birisidir. Kenar belirleme ile görüntüde kenar bulmak için önce görüntü, gri görüntüye dönüştürülmektedir. Gri görüntü üzerindeki gri seviyelerindeki ani değişimler orada bir kenar olabileceğinin işaretidir. Gri görüntü üzerinde kenarları bulmak için parlaklığın aniden değiştiği bölgeler kenar olabilmektedir. Gri seviyeli görüntülerdeki kenarlar, parlaklıkta yer alan süreksizliklerin olduğu bölgeler olarak veya parlaklık derecelerinin belirgin olarak birbirinden farklı olduğu iki bölge arasındaki sınır olarak tanımlabilir. Resim içerisindeki bu değişimler gradyan temelli yöntemler kullanılarak hesap edilmektedir. [6].

Çeşitli kenar belirleme algoritmaları bulunmaktadır. En çok kullanılan kenar belirleme algoritmalar bir sonraki sayfada belirtilmiştir.

- Sobel Kenar Bulma Algoritması
- Prewitt Kenar Bulma Algoritması
- Canny Kenar Bulma Algoritması

2.4.1. Sobel Kenar Bulma Algoritması

Basit matematik modeli ve gerçekleme kolaylığından dolayı görüntü üzerinde kenar bulma işleminde Sobel filtreleme sık kullanılmaktadır. Sobel filtre, öncelikli grupta yer almakta ve görüntünün birinci türevindeki en yüksek ve en düşük olduğu yerlerin tespitiyle kenarların bulunmasını sağlamaktadır. Bu işlem için genellikle 3x3'lük iki tane evrişim matrisi kullanılmaktadır [4].

Şekil 5 de verilen matris y yönünde gradyant büyüklüğünü hesaplamaktadır.

-1	-2	-1
0	0	0
1	2	1

Şekil 5. Y Yönündeki Gradyan Matrisi

Şekil 6 da verilen matris x yönünde gradyant büyüklüğünü hesaplamaktadır.

-1	0	1
-2	0	2
-1	0	1

Şekil 6 X Yönündeki Gradyan Matrisi

Bu matrisler birlikte kullanılarak her bir nokta için mutlak büyüklük hesaplanmaktadır. Bu hesaplama için denklem 2.1'de verilen eşitlik kullanılmaktadır.

$$|G| = \sqrt{G^2{}_X + G^2{}_Y} \tag{2.1}$$

Ancak bir çok uygulamada bu hesaplama denklem 2.2'de olduğu gibi kullanılmaktadır.

$$|G| = |G_X| + |G_Y| \tag{2.2}$$

Denklem 2.2'deki denklem kullanılarak mutlak büyüklük ile Sobel bulma algoritmasının hassasiyet derecesi düşse de hesaplama karmaşıklığını azaltması sebebiyle

tercih edilmektedir. Ortaya çıkan kenarın yön açısı x ve y yönlerindeki değerlerine bakılarak şu şekilde bulunmaktadır [5].

$$\theta = \arctan(\frac{G_Y}{G_X}) \quad (2.3)$$

2.4.2. Prewitt Kenar Bulma Algoritması

Prewit kenar bulma algoritması, Sobel algoritması gibi x ve y ekseninde matrisler kullanarak gradyan hesabı bulup kenarı belirler.Sobel işlecine göre daha çok gürültü içermektedir.[7]

Şekil 7 de verilen matris y yönünde gradyant büyüklüğünü hesaplamaktadır

1	1	1
0	0	0
-1	-1	-1

Şekil 7. Y Yönündeki Gradyan Matrisi

Şekil 8 de verilen matris x yönünde gradyant büyüklüğünü hesaplamaktadır.

-1	0	1
-1	0	1
-1	0	1

Şekil 8. X Yönündeki Gradyan Matrisi

Bu matrisler birlikte kullanılarak tüm noktalar için mutlak büyüklük hesaplanmaktadır. Bu hesaplama için denklem 2.4'de verilen eşitlik kullanılmaktadır.

$$|G| = |G_X| + |G_Y| \quad (2.4)$$

Bu matrisler birlikte kullanılarak tüm noktalar için mutlak büyüklük hesaplanmaktadır.

$$|G| = \sqrt{{G^2}_X + {G^2}_Y} \tag{2.5}$$

Bu hesaplama için denklem 2.6'de verilen eşitlik kullanılmaktadır. Ortaya çıkan kenarın yön açısı x ve y yönlerindeki değerlerine bakılarak şu şekilde bulunmaktadır [8].

$$\theta = \arctan(\frac{G_Y}{G_X}) \tag{2.6}$$

2.4.3. Canny Kenar Belirleme Algoritması

Canny algoritması kenar bulmada çok sık kullanılan algoritmalardan bir tanesidir. Görüntüdeki gürültü Gaussian matrisiyle azaltılıp sonrasında gradyant işlemlerinin yapılmasıyla kenar gradyant büyüklüğü ve yönü hesaplanmaktadır. Kenarlar inceltilir ve görüntüde olmasına gerek duyulmayan gürültüler aşağıdaki metotlarla elimine edilir. Canny kenar belirleme algoritmasının dört adımdan oluşmaktadır [9].

2.4.3.1. Yumuşatma

Bu metotta Gaussian süzgeci yapılmış görüntüye yumuşatma uygulanmaktadır. O[k,m] giriş resmini, G[k,m, σ] Gaussian yumuşatma süzgeci ve σ Gaussian standart sapmasını (yumuşatma derecesini) ifade etmek üzere, orijinal O[k,m] resmi ve G[k,m, σ] Gaussian süzgecinin konvolüsyon işlemi sonucunda elde edilmiş yumuşatma uygulanmış resim Z[k,m] ile sembolize edilmiştir.

$$Z[k,m] = G[k,m,\sigma] * O[k,m] \tag{2.7}$$

2.4.3.2. Gradyan Hesabı

Gradyan büyüklüğü ve gradyan yönü hesaplamaları, türev çözümleri için sonlu fark yaklaşımı ile elde edilmektedir. Önce Z[k,m]'nin kısmi türevleri hesaplanmaktadır.

$$P[k,m] \approx (\,Z[k,m+1] - Z[k,m] + Z[k+1,m+1] - Z[k+1,m])/2 \quad (2.8)$$

$$Q[k,m] \approx (Z[k,m] - Z[k+1,m] + Z[k,m+1] - Z[k+1,m+1])/2 \quad (2.9)$$

X ile Y'nin türevleri 2 satır 2 sütundan oluşan matrisin sonlu farklarının medyanı alınıp hesaplanmaktadır. Buna göre gradyanın büyüklük değeri ve açısı aşağıdaki denklemlerle bulunmaktadır.

$$M(k,m) = \sqrt{P[k,m]^2 + Q[k,m]^2} \quad (2.10)$$

$$Q[k,m] = \arctan(Q[k,m], P[k,m]) \quad (2.11)$$

2.4.3.3. Maksimum Olmayan Noktaların Bastırılması

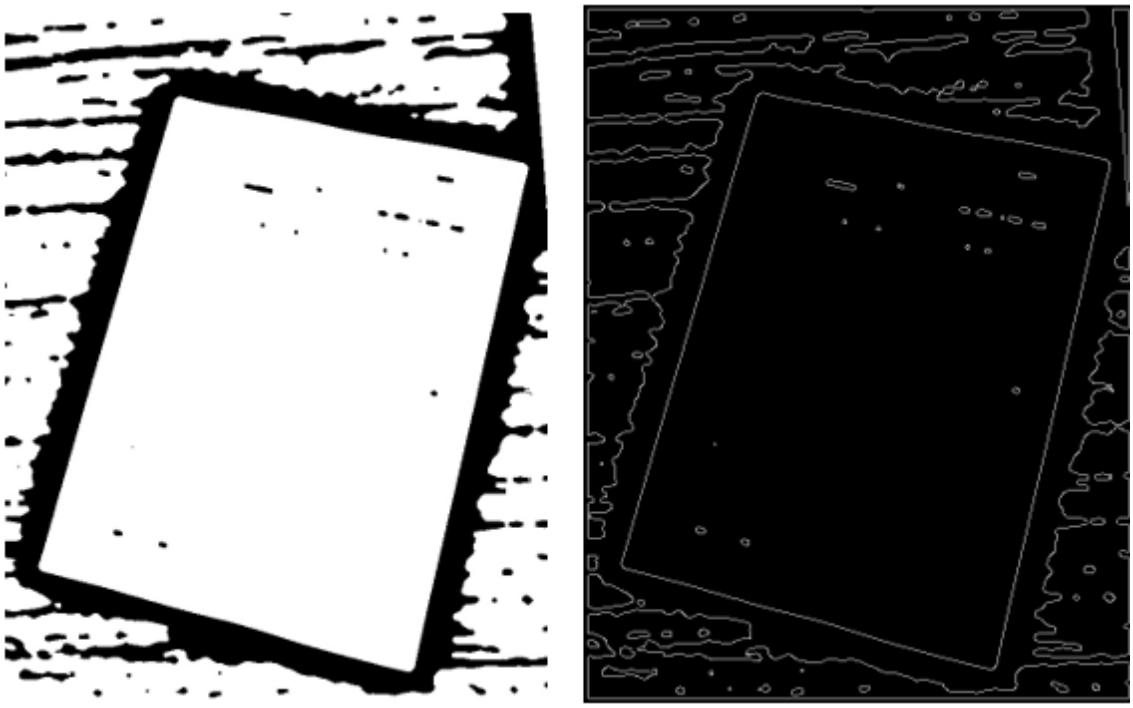
Gradyan algoritması kenar piksellerini belirlemek için gradyanı hesaplamaktadır., Canny algoritması kenarları belirlerken gradyan vektörünün maksimum olduğu noktayı kullanır. Canny algoritması bu özelliği sebebiyle kısıtlayıcıdır. Kenar piksellerinde oluşan kalınlığı gidermek için uygulanan bu yönteme Maksimum Olmayan Noktaların Bastırılması denilmektedir.

$$N[k,m] = nms(M[k,m], C[k,m]) \quad (2.12)$$

2.4.3.4. Eşikleme

Eşikleme yöntemi görüntüdeki kenarların bulunması için kullanılmaktadır. Kenarların daha belirgin olması sağlamak için maksimum olmayan noktaları bastırılmış görüntüye eşikleme metotu uygulanmaktadır. Eşik değerini belirlemek için genellikle bir çok test yapılır ve optimum eşik değeri bulunur. Eşik değer optimum çok daha küçük bir değer seçilecek olursa görüntüdeki gerekesiz kenarlarda bulunabilmektedir. Eşik değer

optimum değerin çok üstünde seçilir ise istenilen kenar kaybedilebilmektedir..



Şekil 9. Canny Kenar Bulma Algoritması Uygulanmış Görüntü

2.5. Konturlama

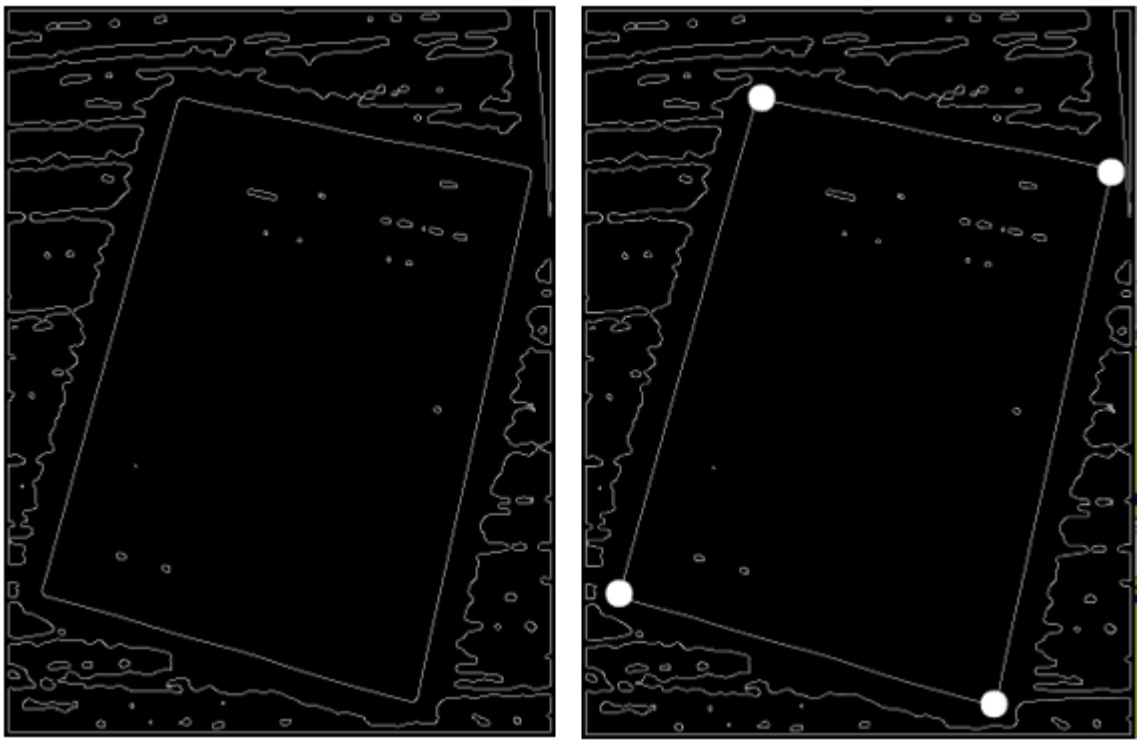
Konturlar, tüm sürekli noktalara katılan bir eğri olarak açıklanabilir. Daha iyi sonuçlar elde etmek için binary görüntüler kullanılmaktadır. Bu nedenle konturlama yapmadan önce kenar algılama algoritmaları kullanmak daha doğru sonuç vermektedir. Bir görüntüye kenar algılama algoritması uygulandıktan sonra konturlama algoritmaları uygulandıktan sonra ortaya görüntüdeki tüm konturların x ve y koordinatlarının olduğu bir dizi çıkmaktadır.

2.5.1. Konturlamada Zincir Kullanılmayan Yaklaşım

Görüntüye Konturlamada Zincir Kullanılmayan Yaklaşım (Chain Approx None) metodu uygulanırsa görüntüdeki tüm konturların sınır noktaları bulunmuş olur. Bu metot kullanılırken algoritmanın ihtiyacının net olarak belirlenmesi gerekmektedir. Çünkü görüntüdeki tüm konturların sınır noktaları belirlendiği için program belleği daha fazla kullanılacaktır. Tüm noktaların kullanılmasına gerekli olmadığında Konturlamada Basit Zincir Yaklaşımı yöntemi kullanılmaktadır.

2.5.2. Konturlamada Basit Zincir Yaklaşımı

Konturlamada Basit Zincir Yaklaşımı (Chain Approx Simple) bu konturlama metodu tüm kontur sınır noktalarının saklanmasının ihtiyaç olmadığında kullanılmaktadır. Bu metot gereksiz tüm noktaları kaldırarak konturu sıkıştırıp daha az noktayla ifade edilebilmesini sağlamaktadır.



Şekil 10. Konturlama Yapılıp Köşe Noktaları Belirlenmiş Görüntü

2.6. Geometrik Görüntü Dönüşümleri

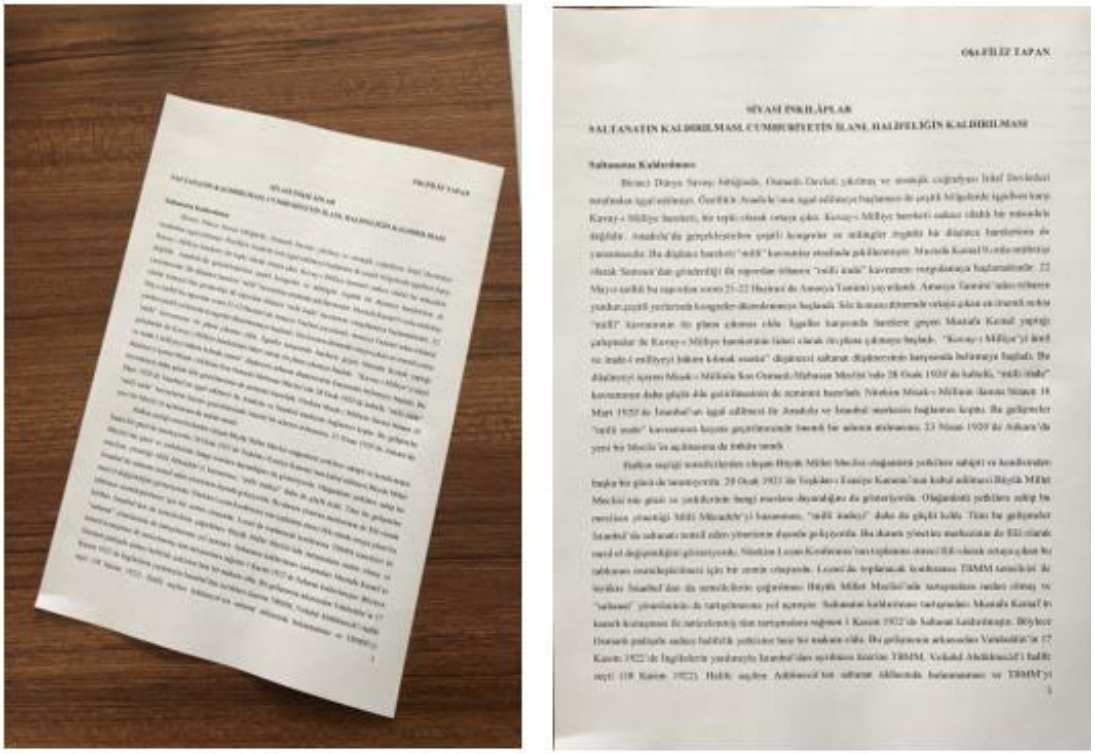
Geometrik görüntü dönüşümleri, görüntünün döndürülmesi, taşınması ve ölçeklenmesi olarak tanımlanmaktadır. Bu çalışma kapsamında da görüntü içerisindeki belge bulunduktan sonra ölçeklenip döndürülmüştür [10].

2.6.1. Affine Dönüşümü

Affine dönüşümü, düzlemleri, noktaları ve düz çizgileri koruyan doğrusal bir haritalama metotudur. Affine dönüşümden sonra paralel çizgi setleri paralel kalır. Affine dönüşüm tekniği tipik olarak ideal olmayan kamera açılarında meydana gelen geometrik bozulmaları veya deformasyonları düzeltmek için kullanılmaktadır. Örneğin, uydu görüntüleri geniş açılı mercek deformasyonunu, panorama dikişini ve görüntü kaydını düzeltmek için affine dönüşümleri kullanılmaktadır. Bozulmayı ortadan kaldırmak için görüntüleri büyük, düz bir koordinat sistemine dönüştürmek ve birleştirmek arzu edilir. Bu, görüntü bozulması için muhasebe gerektirmeyen daha kolay etkileşimler ve hesaplamalar yapılmasını sağlamaktadır.

2.6.2. Perspektif Dönüşümü

En sık kullanılan yöntemlerden biri Perspektif dönüşümdür. Bu metot genellikle birbirine paralel olmayan görüntülerde kullanılmaktadır. Bu metot ile görüntü istenilen koordinatlara taşınabilmektedir. Bu algoritmanın uygulanabilmesi giriş resminden 4 noktaya ihtiyaç vardır. Bu noktalar belgenin köşe noktaları olarak seçilerek 3x3'lük dönüşüm matrisi uygulanarak dönüşüm sağlanmaktadır.



OKAN FELİZ TAPAN

SİYASİ İNKILÂPLAR

SALTANATIN KALDIRILMASI, CUMHURİYETİN İLANI, HALİFELİĞİN KALDIRILMASI

Saltanatın Kaldırılması

Birinci Dünya Savaşı felâketinde, Osmanlı Devleti yıkılmış ve stratejik coğrafyası İtilâf Devletleri tarafından işgal edilmişti. Özellikle Anadolu'nun işgal edilmesiyle başlamış ilk şeytlik bilgilerinde işgallere karşı Kuvay-ı Milliye hareketi, bir teşkilâtlanmak ortaya çıktı. Kuvay-ı Milliye hareketi sadece silâhlı bir mücadele değildir. Anadolu'da gerçekleştirilen yapılı kongreler ve mitingler örgütlü bir düşünce hareketinin de yansımasıdır. Bu düşünce hareketi "milli" kavramına etrafında şekillenmiştir. Mustafa Kemal'in ordu müfettişi olarak Samsun'dan gönderildiği ilk raporlarda itibaren "milli irade" kavramını vurgulamaya başlamalarıdır. 22 Mayıs tarihli bu raporlar sonra 21-22 Haziran'da Amasya Tamimi yayınlandı. Amasya Tamimi'nden itibaren yürütülen çeşitli yerlerinde kongreler düzenlenmeye başlandı. Söz konusu dönemde ortaya çıkan en önemli nokta "milli" kavramının ön plana çıkması oldu. İşgallere karşısında hareketin gayesi Mustafa Kemal yaptığı çalışmalar ile Kuvay-ı Milliye hareketinin lideri olarak ön plana çıkmaya başladı. "Kuvay-ı Milliye"yi ümit ve irade-i milliyeyi bütün kılmak esastır" diğüncesi saltanat düşüncesinin karşısında belirmeye başladı. Bu düşünceyi aynen Misak-ı Milli'nin Son Osmanlı Mebusan Meclisi'nde 28 Ocak 1920 de kabulü, "milli irade" kavramının daha çözülü ifade gelebilmesinin de zeminini hazırladı. Nitekim Misak-ı Milli'nin ilanını Mütarek 16 Mart 1920'de İstanbul'un işgal edilmesi ile Anadolu ve İstanbul merkezinin bağlantısı koptu. Bu gelişmeler "milli irade" kavramının hayata geçirilmesinde önemli bir adımın atılmasını, 23 Nisan 1920'de Ankara'da yeni bir Meclis'in açılmasını da ifade etmiş

Padişah saygılı temsilcilerden oluşan Büyük Millet Meclisi oluşmasını şekillere sahipti ve kendisinden başka bir güçlü de tanımıyordu. 20 Ocak 1921'de Teşkilât-ı Esasiye Kanunu'nun kabul edilmesi Büyük Millet Meclisi'nin gücü ve yetkilerinin hangi sınırlara dayandığını da göstermiştir. Oluşturduğu şekillere sahip bu meclisin yönetiği Milli Mücadele'yi hazırlamış, "milli iradeyi" daha da güçlü kıldı. Tüm bu gelişmeler İstanbul'da saltanatı temsil eden yönetimin dışında gelişiyordu. Bu durum yönetim merkezinin de Bâb-ı âlînin ayrıl et değişmediğini gösteriyordu. Nitekim Lozan Konferansı'nın toplanma süresi Bâb-ı âlînin ortaya çıkan bir takbirinin sonuçlandırılmasıyla ilgili bir zemin oluşturdu. Lozan'da toplanacak konferansa TBMM temsilcisi ile birlikte İstanbul'dan da temsilcilerin çağrılması Büyük Millet Meclisi'nde tartışmalara neden olmuş ve "saltanat" yönetiminin de tartışılmasına yol açmıştır. Saltanatın kaldırılması tartışmaları Mustafa Kemal'in kararlı konuşması ile sonlandırılmış olan tartışmalara sağlanan 1 Kasım 1922'de Saltanat kaldırılmıştır. Böylece Osmanlı padişahı sadece hukuksal yönünü hem bir makam oldu. Bu gelişmenin arkasından Vahdettin'in 17 Kasım 1922'de İngilizlerin yardımıyla İstanbul'dan ayrılması üzerine TBMM, Vekâlet Abdülmecid'i halife seçti (18 Kasım 1922). Halife seçilen Abdülmecid bir saltanat fetvasında bulunmamıştır ve TBMM'yi

BÖLÜM 3. MATERYAL VE YÖNTEM

Görüntü işleme çalışmalarında farklı programlama dilleri ve farklı kütüphaneler kullanılmaktadır. En çok kullanılan programlama diller C++, Python, Matlab olmasına karşın farklı programlama dilleride yer almaktadır. Bu çalışma Python programlama dili ile kodlanmıştır. Python, kararlı ve hızlı çalışa çok sayıda kütüphanesinin olması, çok sayıda dökümana kolayca erişilebilmesi ve günümüzde en çok kullanılan programlama dillerinden biri olması sebebiyle tercih edilmiştir.

Bu projede kullanılan donanım özellikleri ise Intel Core i7-7500 CPU 2.7 Ghz işlemci, 8 gb fiziksel 16 gb sanal ram bellekli bir bilgisayar ve Python 3.7 sürümü kullanılmıştır.

3.1. Anaconda

Anaconda içerisinde bir on binlerce açık kaynak kodlu paket içeren bir Python platformudur. Kurulumları yapıldıktan sonra içerisinde Spyder gibi Python kodlarını yazıp derleyebileceğiniz programlar bulumaktadır. Anaconda'nın dünya çapında sıkça kullanılmasının birkaç sebebi vardır . Bunlar ;

2. İstenilen Python sürümünü kurup kullanmanıza olanak sağlar.
3. Performans olarak yüksek bir verimlilik sağlar.
4. Bir pakette değişiklik yapmak veya yüklemek için yönetici yetkisine gerek duymaz. Bu sebeple çalışmayı daha kolay getirir.
5. Windows, Linux ve MacOS işletim sistemleri üzerinde çalışabilmesi en büyük artılarıdır.

3.2. OpenCV Kütüphanesi

OpenCV, bilgisayarla görme, makine öğrenimi ve görüntü işleme için büyük bir açık kaynak kütüphanesidir ve günümüzün sistemlerinde çok önemli olan gerçek zamanlı operasyonda önemli bir rol oynamaktadır. Bunu kullanarak, bir insanın nesnelerini,

yüzlerini ve hatta el yazısını tanımlamak için görüntüler ve videolar işlenebilmektedir. Numpy gibi çeşitli kitaplıklarla entegre edildiğinde, python OpenCV dizi yapısını analiz için işleyebilmektedir. Görüntü desenini ve çeşitli özelliklerini tanımlamak için vektör uzayını kullanılır ve bu özellikler üzerinde matematiksel işlemler yapılmaktadır.

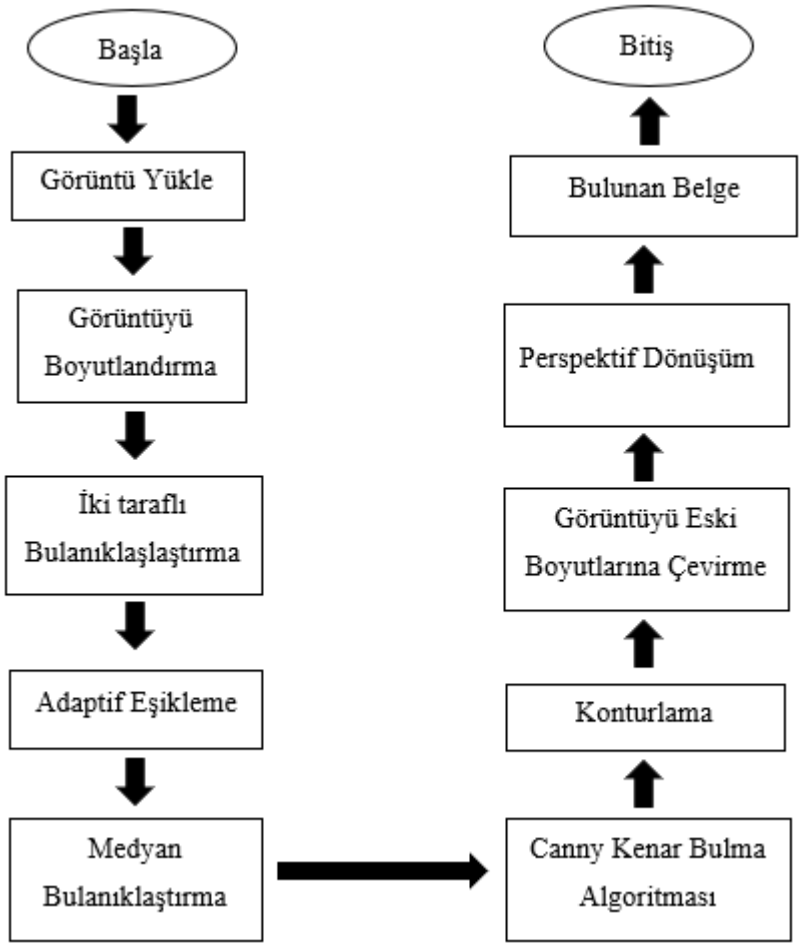
OpenCV bir BSD lisansı altında piyasaya sürülmüş ve bu nedenle hem akademik hem de ticari kullanım için ücretsiz bir kütüphane olmuştur. C ++, C, Python ve Java arayüzlerine sahiptir ve Windows, Linux, Mac OS, iOS ve Android'i desteklemektedir..

BÖLÜM 4. GERÇEKLEŞTİRİLEN UYGULAMA

Bu bölümde taranmış veya fotoğrafı çekilmiş görüntü içerisindeki eğik belgenin bulunup çıkarılmasının akış şeması ile görüntüye uygulanan aşamalar sonucundaki değişimlerin gösterildiği şekiller verilmiştir.

4.1. Uygulama Akış Şeması

Uygulamayı kodlama aşamasına geçmeden önce aşağıdaki akış şeması çıkartılarak tasarlanmış ve kodlama aşamasına geçilmiştir.

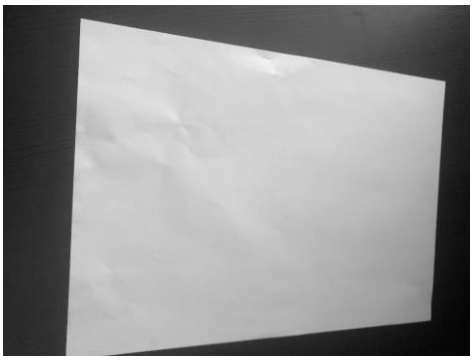


Şekil 12. Uygulama Akış Şeması

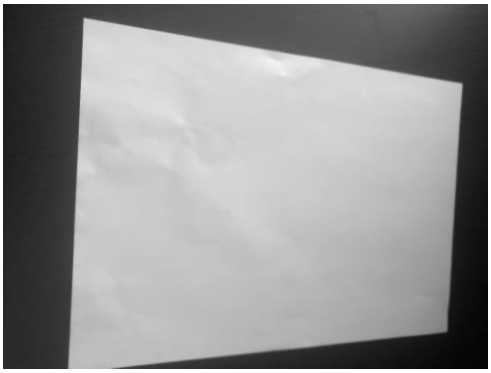
4.2 Görüntüye Uygulanan İşlemler

Örnek bir görüntü üzerinde yapılan işlemlerin adım adım gösterildiği şekiller altta paylaşılmıştır. Böylelikle uygulamanın resimler üzerinde ne gibi değişimler yaptığı görüntülenebilmektedir.

Şekil 13. Orijinal Görüntü

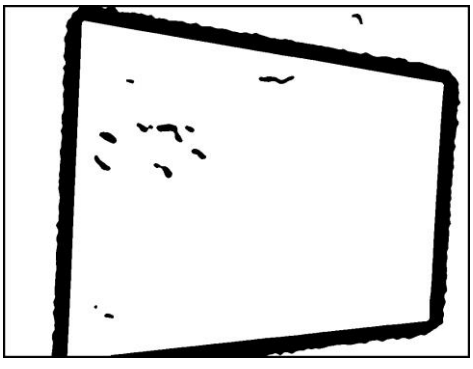


Şekil 14. Gri Görüntü



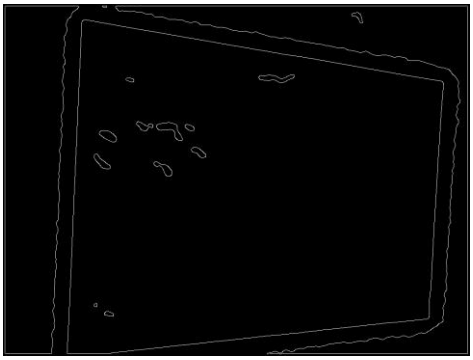
Şekil 15. İki Taraflı Bulanıklaştırma

Şekil 16. Adaptif Filtreleme

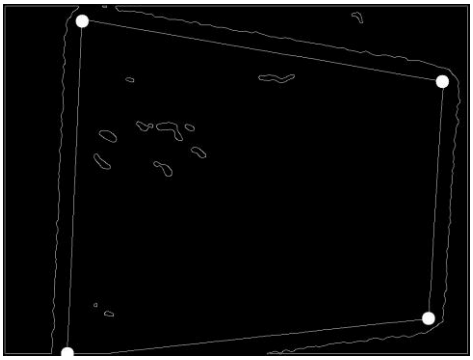


Şekil 17. Kenardakı Boslukları Giderme

Şekil 18. Medyan Bulanıklaştırma



Şekil 19. Canny Kenar Bulma Algoritması



Şekil 20. 4 Köşe Noktalarının Bulunması

Şekil 21. Bulunmuş Belge Görüntüsü

Böylelikle giriş resmi olarak verilmiş taranmış veya fotoğrafı çekilmiş görüntünün çeşitli filtre ve algoritmalar kullanılarak adım adım ne gibi aşamalardan geçtiği görülmüş ve de istenen belge görüntü içerisinden bulunup çıkartılmıştır.

BÖLÜM 5. SONUÇ

Giriş bölümünde de bahsedildiği üzere tanımlanan problem, taranmış veya fotoğrafı çekilmiş bir belge dökümanının görüntü içerisinde eğik durmasıdır. Fiziksel ortamda dijital ortama bu görüntüleri aktarırken gereksiz detayların elimine edilmesi ve görüntüde belge dökümanının yer alması amaçlanmıştır. Bu amaçlar doğrultsunda programa arka plan rengi belge dökümanından farklı renkte olmak koşuluyla bir görüntü verilmiştir. Giriş olarak alınan bu görüntü önce gri tonlama uygulanarak gri görüntüye çevrilmiştir. Elde edilen gri görüntü üzerinde bilatereal dönüşüm uygulanıp gürültü azaltılmış kenarlar keskinleştirilmiştir. Bilatereal görüntü üzerinde Adaptif Eşikleme yöntemi uygulanarak görüntü binary görüntüye çevrilmiştir. Binary görüntüye Medyan Bulanıklaştırma yöntemi uygulanarak gereksiz küçük detaylardan elimine edilmiştir. Sonrasında bu görüntüye Canny Kenar Bulma Algoritması uygulanarak kenarlar elde edilmiştir. Görüntü içerisinde kenarlar belirlendikten sonra Konturlama yapılarak dökümanın köşe noktaları bulunmuştur. Bulunan köşeler yardımıyla artık belgenin boyu eni gibi bilgileri Öklid methoduyla ulaşılmıştır. Bu sayede ortaya çıkacak yeni görüntü en ve boy bilgisi bulunmuştur. Köşe noktaları bilinen orijinal resim üzerinde Perspektif dönüşüm uygulanarak görüntü içerisinde belge dökümanı çıkarılmış ve yeni görüntü olarak kaydedilmiştir.

EKLER

EK – 1. Uygulama Kaynak Kodları

############## Eğik Belge Bulma ve Düzeltme Programı#################

Her aşamadan sonra görüntüyü kontrol etmek için dosyaya kaydedilmiştir.

```
#Kütüphane tanimlari

import numpy as np
import matplotlib.pyplot as plt
import cv2

# Görüntüyü yükleme.

image =
cv2.cvtColor(cv2.imread("C:/Users/pc/Desktop/Tez/2.jpg"),cv2.COLOR_BGR2RGB)
#### Resmin boyunu ayarlamak için resize fonksiyonu #####

def resize(img, height=800):
    rat = height / img.shape[0]
    return cv2.resize(img, (int(rat * img.shape[1]), height))

### Grayscale'e dönüşüm ###

img = cv2.cvtColor(resize(image), cv2.COLOR_BGR2GRAY)

### Gri görüntüyü dosyaya kaydet ###
cv2.imwrite("C:/Users/pc/Desktop/Tez/gri_goruntu.jpg",          cv2.cvtColor(img,
cv2.COLOR_BGR2RGB))

### Bilateral filtreleme ###
```

```
img = cv2.bilateralFilter(img, 9, 75, 75)

### Bilateral Görüntüyü dosyaya kaydet ###

cv2.imwrite("C:/Users/pc/Desktop/Tez/bilateral_goruntu.jpg", cv2.cvtColor(img, cv2.COLOR_BGR2RGB))

### Threshold'a göre adaptif resim oluşturma ###

img = cv2.adaptiveThreshold(img, 255, cv2.ADAPTIVE_THRESH_GAUSSIAN_C, cv2.THRESH_BINARY, 115, 4)

### Adaptif resmi kaydet ###
cv2.imwrite("C:/Users/pc/Desktop/Tez/adaptive_goruntu.jpg", cv2.cvtColor(img, cv2.COLOR_BGR2RGB))


###  Median filtreleme ile küçük detaylarin elimine edilmesi ###

img = cv2.medianBlur(img, 11)

### Median görüntüyü kaydet ###
cv2.imwrite("C:/Users/pc/Desktop/Tez/median_blur_goruntu.jpg", cv2.cvtColor(img, cv2.COLOR_BGR2RGB))

###  Sayfanin kenarlarindaki bosluklari doldurmak için 5 px'lik ekleme yapma ###

img = cv2.copyMakeBorder(img, 5, 5, 5, 5, cv2.BORDER_CONSTANT, value=[0, 0, 0])

### Kenar boşluklarının doldurulduğu görüntü ###
cv2.imwrite("C:/Users/pc/Desktop/Tez/bosluk_giderme_goruntu.jpg", cv2.cvtColor(img, cv2.COLOR_BGR2RGB))
```

```
### Canny Kenar Algoritmasi ###
edges = cv2.Canny(img, 200, 250)

### Canny uygulanmış Görüntü ###
cv2.imwrite("C:/Users/pc/Desktop/Tez/canny_goruntu.jpg", cv2.cvtColor(edges, cv2.COLOR_BGR2RGB))

# Countourlama
contours, hierarchy = cv2.findContours(edges, cv2.RETR_TREE, cv2.CHAIN_APPROX_SIMPLE)
height = edges.shape[0]
width = edges.shape[1]
MAX_COUNTOUR_AREA = (width - 10) * (height - 10)
maxAreaFound = MAX_COUNTOUR_AREA * 0.5
pageContour = np.array([[5, 5], [5, height-5], [width-5, height-5], [width-5, 5]])


for cnt in contours:
    perimeter = cv2.arcLength(cnt, True)
    approx = cv2.approxPolyDP(cnt, 0.03 * perimeter, True)

    # Sayfanin 4 köse noktalari için

    if (len(approx) == 4 and cv2.isContourConvex(approx) and maxAreaFound < cv2.contourArea(approx) < MAX_COUNTOUR_AREA):

        maxAreaFound = cv2.contourArea(approx)
        pageContour = approx

########################################
# köşe noktalarını göster
#cv2.drawContours(edges, pageContour, -1, (255,0,0), 30)
```

```
#cv2.imshow("title", edges)
#cv2.waitKey(10000)
########################################

def fourCornersSort(pts):
    """ Köselerin sol üst,sag alt, sol alt, sag üst diye ayirimi"""

    diff = np.diff(pts, axis=1)
    summ = pts.sum(axis=1)

    # np.argmin() en küçük olan indisli elemani döndürür.
    return
np.array([pts[np.argmin(summ)],pts[np.argmax(diff)],pts[np.argmax(summ)],pts[np.argm
in(diff)]])



def contourOffset(cnt, offset):
    """ Countourlamada verilen 5piksellik fazlaligin geri alinmasi. """

    cnt += offset
    cnt[cnt < 0] = 0
    return cnt

pageContour = fourCornersSort(pageContour[:, 0])
pageContour = contourOffset(pageContour, (-5, -5))

###  Görüntüyü tekrar scale etmek için ###
sPoints = pageContour.dot(image.shape[0] / 800)


### Euclidean uzaklik formülüyle belgenin kenar uzunluklarını hesaplama###
```

```
height = max(np.linalg.norm(sPoints[0] - sPoints[1]),np.linalg.norm(sPoints[2] - sPoints[3]))
width = max(np.linalg.norm(sPoints[1] - sPoints[2]),np.linalg.norm(sPoints[3] - sPoints[0]))

tPoints = np.array([[0, 0],[0, height],[width, height],[width, 0]], np.float32)

###  PerspectiveTransform() için float32 dönüsümü yapilmalidir. ###

if sPoints.dtype != np.float32:
   sPoints = sPoints.astype(np.float32)

M = cv2.getPerspectiveTransform(sPoints, tPoints)
newImage = cv2.warpPerspective(image, M, (int(width), int(height)))

### yeni olusan resmin renklendirilip kaydedilmesi. ###
cv2.imwrite("C:/Users/pc/Desktop/Tez/yeni_goruntu.jpg", cv2.cvtColor(newImage, cv2.COLOR_BGR2RGB))
```

KAYNAKÇA

[1] Rouse, Margaret. Grayscale. “https://whatis.techtarget.com/definition/grayscale”, Erişim Tarihi: 10.05.2020

[2] Szeliski, Richard. Computer Vision: Algorithms and Applications. “https://docs.opencv.org/2.4/doc/tutorials/imgproc/gausian_median_blur_bilateral_filter/ ”gausian_median_blur_bilateral_filter.html”, Erişim Tarihi: 12.05.2020

[3] Szeliski, Richard. Computer Vision: Algorithms and Applications. “https://docs.opencv.org/3.4/d7/d4d/tutorial_py_thresholding.html” , Erişim Tarihi: 13.05.2020

[4] Onat, Emre. Sobel Filtre Kullanarak Gerçek Zamanlı Video Sinyallerinin İşlenmesinin FPGA’da Gerçeklenmesi, “https://www.researchgate.net/publication/317105904_Sobel_Filtre_Kullanarak_Gercek_Zamanli_Video_Sinyallerinin_Islenmesinin_FPGA'da_Gerceklenmesi” , Erişim Tarihi: 15.05.2020

[5] Sundari, V. K., Manikandan, M., Prakash, P., "FPGA Implementation of Sobel Edge Detector", IJAST - International Journal of Advances in Science and Technology , Erişim Tarihi: 15.05.2020

[6] Aybar, Elif. Sobel İşlei Kullanılarak Renklei Görüntülerde Kenar Bulma. “https://fenbildergi.aku.edu.tr/pdf/0801/8-1(217-230).pdf”, Erişim Tarihi: 15.05.2020

[7] Tafralı, Merve, Kenar Belirleme. “https://gist.github.com/mervetafrali/69f55d66d168b61cce84fbb91157e43b” , Erişim Tarihi: 17.05.2020

[8] Cayiroglu, İbrahim. Kenar Bulma Algoritmaları "http://www.ibrahimcayiroglu.com/Dokumanlar/GoruntuIsleme/Goruntu_Isleme_Ders_Notlari-7.Hafta.pdf" , Erişim Tarihi: 17.05.2020

[9] Turan, Başar. Görüntü İşleme Algoritmalarının Eş Zamanlı Süreçlere Ayrılarak Kablosuz Ağ Üzerinden Gerçeklenmesi ve Performans Analizleri. "http://openaccess.maltepe.edu.tr/xmlui/bitstream/handle/20.500.12415/3725/486690.pdf?sequence=1&isAllowed=y" , Erişim Tarihi: 20.05.2020

[10] Szeliski, Richard. Computer Vision: Algorithms and Applications, Erişim Tarihi:.23.05.2020

[11] Pişkin, Mesut Opencv "http://mesutpiskin.com/blog/opencv-nedir.html " , Erişim Tarihi: 25.05.2020