Görüntü İşleme Uygulamaları ile İlgili Çalışma Raporu

Bilgisayar Mühendisliği

Ömer Burak ÖZGÜR

omerburakozgur1@gmail.com

194410013

16.01.2022

İçindekiler

1.1) Görüntü İşlemenin Tanımı

Görüntü İşleme: Askeri, endüstri, robotik, astronomi, tıp, coğrafya gibi günlük yaşamın pek çok alanında kullanılan, alınan görüntünün işlenmesinden sonra devamında yapay sinir ağları (YSA), bulanık mantık gibi pek çok algoritma ile değerlendirilen bir teknolojidir. Günümüzde artık her işlemin insan yerine otonom sistemlere yaptırılması, görüntü işlemenin önemini bir kat daha artırmıştır.

Görüntü Nedir: Görüntü 3 Boyutlu nesnelerin 2 Boyutlu yüzey üzerine düşürülmüş haritası olarak tanımlanabilir. Bu haritalamada her noktanın konum bilgisi f(x, y) ve renk bilgisi tutulur. Bu resimlerin insan gözünün görebileceği şekilde gösterilmesine ise Resim diyebiliriz. Görüntü henüz daha sinyal şeklinde ise, insan gözünün görebileceği şekilde değilse resim haline gelmemiş demektir.

Görüntü **Analog** ve **Dijital** görüntü olarak iki kısımda ele alınabilir.

1.2) Görüntü İşlemenin Kullanıldığı Alanlar

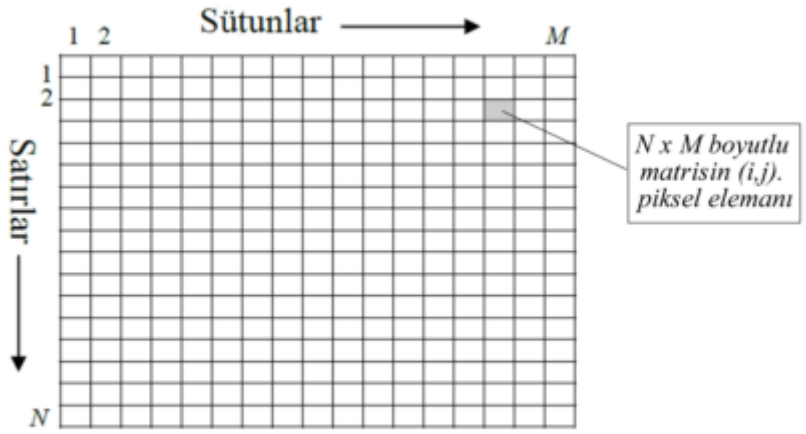
Resim video çekme uygulamaları, Ekran görüntülemeleri ve Yazdırma işlemleri, Resim saklama ve iletimini kolaylaştırmak, Güvenlik uygulamaları, Bilgi çıkarma, Otonom görüş yeteneği, Sanal gerçeklik uygulamaları görüntü işlemenin insanların gündelik hayatında kullanıldığı alanlara örnek verilebilir.

1.3) Görüntü Örnekleme (Sampling)

Analog bir görüntünün bilgisayarın anlayabileceği dijital bir yapıya dönüştürülmesi görüntü örnekleme işlemi ile mümkün olur. Sayısal bir görüntü, sürekli bir görüntü fonksiyonu üzerinden eşit aralıklarla x-ekseni boyunca N adet örnek ve y ekseni boyunca M adet örnek alınarak oluşturulabilir. Böylece, sürekli-zamanlı görüntü fonksiyonundan ayrık zamanlı görüntü fonksiyonuna geçiş gerçekleşmiş olur. 2-B ayrık-zamanda yatayda N ve düşeyde M örnekten oluşan toplam N × M sonlu örnek değeri ile analog bir görüntü yaklaşık olarak ifade edilebilir. Bu işlemde, analog görüntü fonksiyonu düzgün örneklenmiş olur. Yani düzgün örnekleme, analog görüntüden hem yatay hem de düşey yönde eşit aralıklarla örnek alınarak oluşturulur. Oluşan dijital (sayısal) görüntü aslında N satır ve M sütundan oluşan bir matrisdir.

1.4) Görüntü Nicemleme

Görüntünün her bir elemanın (pikselin) parlaklık şiddetini gösteren pozitif tamsayı değeri, Nicemleme ile belirlenir. Görüntü elemanının en küçük ve en büyük genlik değerleri aralığı basamaklara ayrılarak, ilgili basamak değerine en yakın olan görüntü değerini almasıdır. Yani analog renk değerinin, geçerli en yakın dijital renk değerine dönüştürülmesidir. Bu iki işlem sonucunda, bilgisayarlar tarafından işlenebilen sayısal bir görüntü elde edilmiş olur.



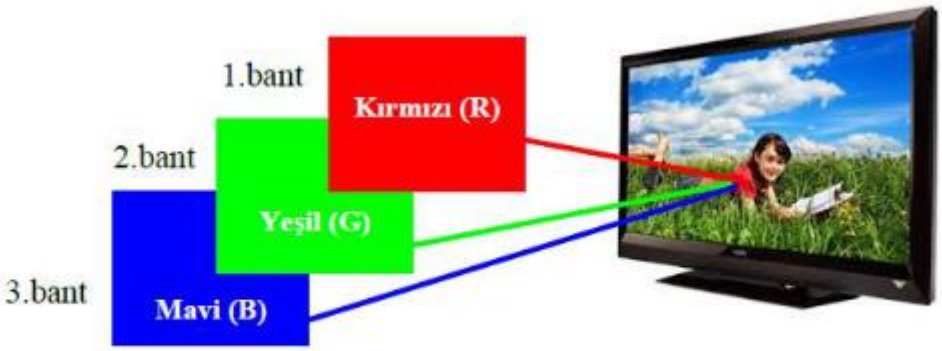
Şekil. N × M büyüklüğüne sahip 2-B bir sayısal görüntünün temel yapısı.

1.5) Gri Görüntü

Sayısallaştırma işleminde, görüntü boyutlarının ve her bir pikselin sahip olabileceği parlaklık değerinin belirlenmesi gerekir. Sayısal görüntünün her bir pikselinin sahip olduğu parlaklık değeri gri seviyeler olarak adlandırılır. Her bir pikseldeki parlaklık değerinin kodlandığı bit sayısına göre gri seviye aralığı belirlenir. Gri seviye sınırlarında iki renk vardır, siyah ve beyaz. Bu ikisi arasında kodlanan görüntülere ise gri-ton (grayscale, monochromatic) görüntüler adı verilir.

1.6) Renkli Görüntü

Renkli görüntüler, R(Kırmızı), G(Yeşil), B (Mavi) kodlanmış aynı cisme ait üç adet gri düzeyli görüntünün üst üste ekranda gösterilmesi ile oluşur. Renkli görüntüyü oluşturan bu üç renk bant olarak isimlendirilir.



Şekil. Renkli görüntünün bilgisayar ekranındaki oluşumu.

2.1) Görüntü Yükleme ve Kaydetme

Yeni görüntülerin de program üzerinde kullanılıp test edilmesi ve işlenebilmesi için görüntü yükleme ve kaydetme özelliği eklendi. Bu sayede kullanıcı istediği gibi istediği görüntü üzerinde işlem yapabilir ve kaydedebilir.

Dosya Açma

```
public void DosyaAc()
        {

            try // Kullanıcı dosya seçme ekranında dosya seçmezse diye hata kontrolü
yapıyoruz.
            {
                openFileDialog1.DefaultExt = ".jpg"; //Default dosya tipi.

                openFileDialog1.Filter = "Image
Files(*.BMP;*.JPG;*.GIF)|*.BMP;*.JPG;*.GIF|All files (*.*)|*.*"; //Gösterilecek dosya
tiplerinin filtrelenmesi.

                openFileDialog1.ShowDialog(); //Dosya seçme ekranının kullanıcıya
gösterilmesi.

                String dosyaYolu = openFileDialog1.FileName; //Dosya yolunun değişkene
aktarılması.
                pictureBox1.Image = Image.FromFile(dosyaYolu); //Kullanıcıdan alının
fotoğrafın picture box üzerinde gösterilmesi.

            }

            catch
            {

            }
        }
```

Çıkış Görüntüsünü Kaydetme

```
public void cikisGoruntusunuKaydet()
        {
            SaveFileDialog saveFileDialog1 = new SaveFileDialog();
            saveFileDialog1.Filter = "Jpeg Resmi|*.jpg|Bitmap Resmi|*.bmp|Gif
Resmi|*.gif";


            saveFileDialog1.Title = "Görüntüyü Kaydet";
            saveFileDialog1.ShowDialog();

            if (saveFileDialog1.FileName != "") //Dosya adı boş değilse kaydedecek.
            {
                // FileStream nesnesi ile kayıtı gerçekleştirecek.
                FileStream fileStream = (FileStream)saveFileDialog1.OpenFile();

                switch (saveFileDialog1.FilterIndex)
                {
                    case 1:
                        pictureBox2.Image.Save(fileStream,
System.Drawing.Imaging.ImageFormat.Jpeg);
                        break;

                    case 2:
                        pictureBox2.Image.Save(fileStream,
System.Drawing.Imaging.ImageFormat.Bmp);
                        break;

                    case 3:
                        pictureBox2.Image.Save(fileStream,
System.Drawing.Imaging.ImageFormat.Gif);
                        break;
                }

                fileStream.Close();
            }
        }
```

3) Görüntü İşleme İşlemleri

3.1) Parlaklık Değerini Arttırma

```
public void ResminParlakliginiArttir()
        {
            int R = 0, G = 0, B = 0;
            Color OkunanRenk, DonusenRenk;
            Bitmap GirisResmi, CikisResmi;
            GirisResmi = new Bitmap(pictureBox1.Image);

            int ResimGenisligi = GirisResmi.Width; //GirisResmi global tanımlandı.
            int ResimYuksekligi = GirisResmi.Height;
            CikisResmi = new Bitmap(ResimGenisligi, ResimYuksekligi); //Cikis resmini
oluşturuyor. Boyutları giriş resmi ile aynı olur.

            int i = 0, j = 0; //Çıkış resminin x ve y si olacak.
            for (int x = 0; x < ResimGenisligi; x++)
            {
                j = 0;
                for (int y = 0; y < ResimYuksekligi; y++)
                {
                    OkunanRenk = GirisResmi.GetPixel(x, y);
```

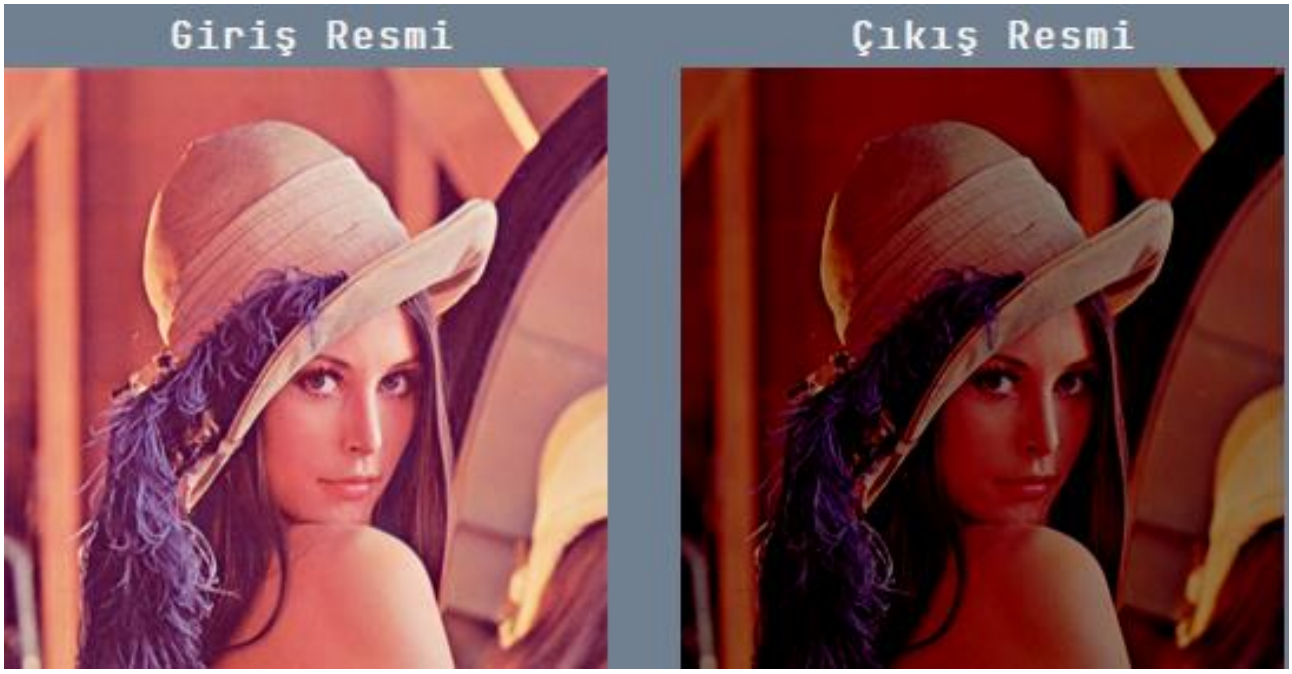
```
//Rengini kullanıcıdan alınan parlaklık değeri ile arttırılacak. //Kullanıcı negatif girer
diye mutlak değere aldık.

                    R = OkunanRenk.R + Math.Abs(parlaklikDegeri);
                    G = OkunanRenk.G + Math.Abs(parlaklikDegeri);
                    B = OkunanRenk.B + Math.Abs(parlaklikDegeri);

                    //Renkler 255 geçtiyse son sınır olan 255 alınacak.
                    if (R > 255) R = 255;
                    if (G > 255) G = 255;
                    if (B > 255) B = 255;

                    if (R < 0) R = 0;
                    if (G < 0) G = 0;
                    if (B < 0) B = 0;

                    DonusenRenk = Color.FromArgb(R, G, B);
                    CikisResmi.SetPixel(i, j, DonusenRenk);
                    j++;
                }
                i++;
            }
            pictureBox2.Image = CikisResmi;
                                                }
```

Görüntünün 100 değeri ile parlaklığının arttırılması.

3.2) Parlaklık Değerini Azaltma

```
public void ResminParlakliginiAzalt()
        {
            int R = 0, G = 0, B = 0;
            Color OkunanRenk, DonusenRenk;
            Bitmap GirisResmi, CikisResmi;
            GirisResmi = new Bitmap(pictureBox1.Image);

            int ResimGenisligi = GirisResmi.Width; //GirisResmi global tanımlandı.
            int ResimYuksekligi = GirisResmi.Height;
            CikisResmi = new Bitmap(ResimGenisligi, ResimYuksekligi); //Cikis resmini
oluşturuyor. Boyutları giriş resmi ile aynı olur.

            int i = 0, j = 0; //Çıkış resminin x ve y si olacak.
            for (int x = 0; x < ResimGenisligi; x++)
            {   j = 0;
```

```
                for (int y = 0; y < ResimYuksekligi; y++)
                {
                 OkunanRenk = GirisResmi.GetPixel(x, y);

                    //Rengini kullanıcıdan alınan parlaklık değeri ile arttırılacak.
                    R = OkunanRenk.R - Math.Abs(parlaklikDegeri);
                    G = OkunanRenk.G - Math.Abs(parlaklikDegeri);
                    B = OkunanRenk.B - Math.Abs(parlaklikDegeri);

                    //Renkler 255 geçtiyse son sınır olan 255 alınacak.
                    if (R > 255) R = 255;
                    if (G > 255) G = 255;
                    if (B > 255) B = 255;

                    if (R < 0) R = 0;
                    if (G < 0) G = 0;
                    if (B < 0) B = 0;

                    DonusenRenk = Color.FromArgb(R, G, B);
                    CikisResmi.SetPixel(i, j, DonusenRenk);
                    j++;
                }
                i++;
            }
            pictureBox2.Image = CikisResmi;
        }
```

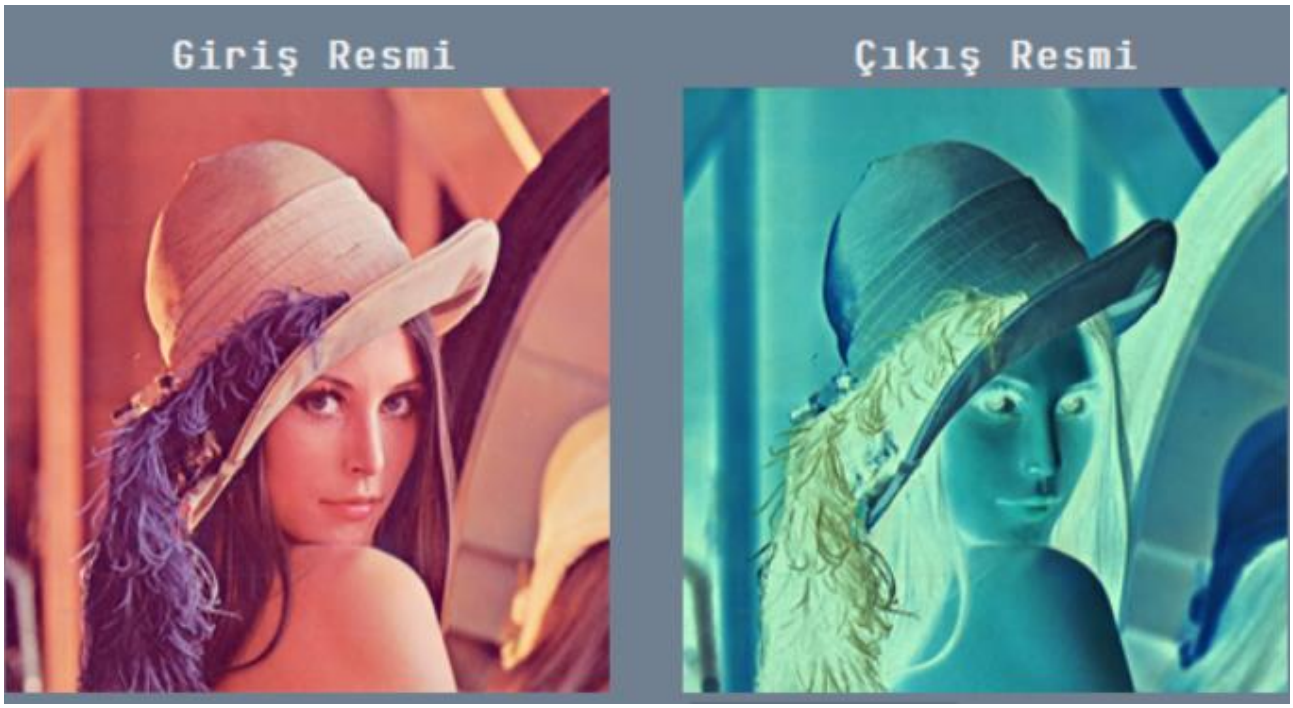


Görüntünün 100 değeri ile parlaklığının azaltılması.

3.3) Görüntünün Negatifini Alma

```
private void negatifOlarakKaydet()
        {
            Color OkunanRenk, DonusenRenk;
            int R = 0, G = 0, B = 0;

            Bitmap GirisResmi, CikisResmi;
            GirisResmi = new Bitmap(pictureBox1.Image);


            ResimGenisligi = GirisResmi.Width; //GirisResmi global tanımlandı. İçerisine
görüntü yüklendi.
            ResimYuksekligi = GirisResmi.Height;
            CikisResmi = new Bitmap(ResimGenisligi, ResimYuksekligi); //Cikis resmini
oluşturuyor.Boyutları giriş resmi ile aynı olur. Tanımlaması globalde yapıldı.
```

```
            int i = 0, j = 0; //Çıkış resminin x ve y si olacak.
            for (int x = 0; x < ResimGenisligi; x++)
            {
                for (int y = 0; y < ResimYuksekligi; y++)
                {
                    OkunanRenk = GirisResmi.GetPixel(x, y);

                    R = 255 - OkunanRenk.R;
                    G = 255 - OkunanRenk.G;
                    B = 255 - OkunanRenk.B;

                    DonusenRenk = Color.FromArgb(R, G, B);
                    CikisResmi.SetPixel(x, y, DonusenRenk);
                }
            }

            pictureBox2.Image = CikisResmi;

           }
```

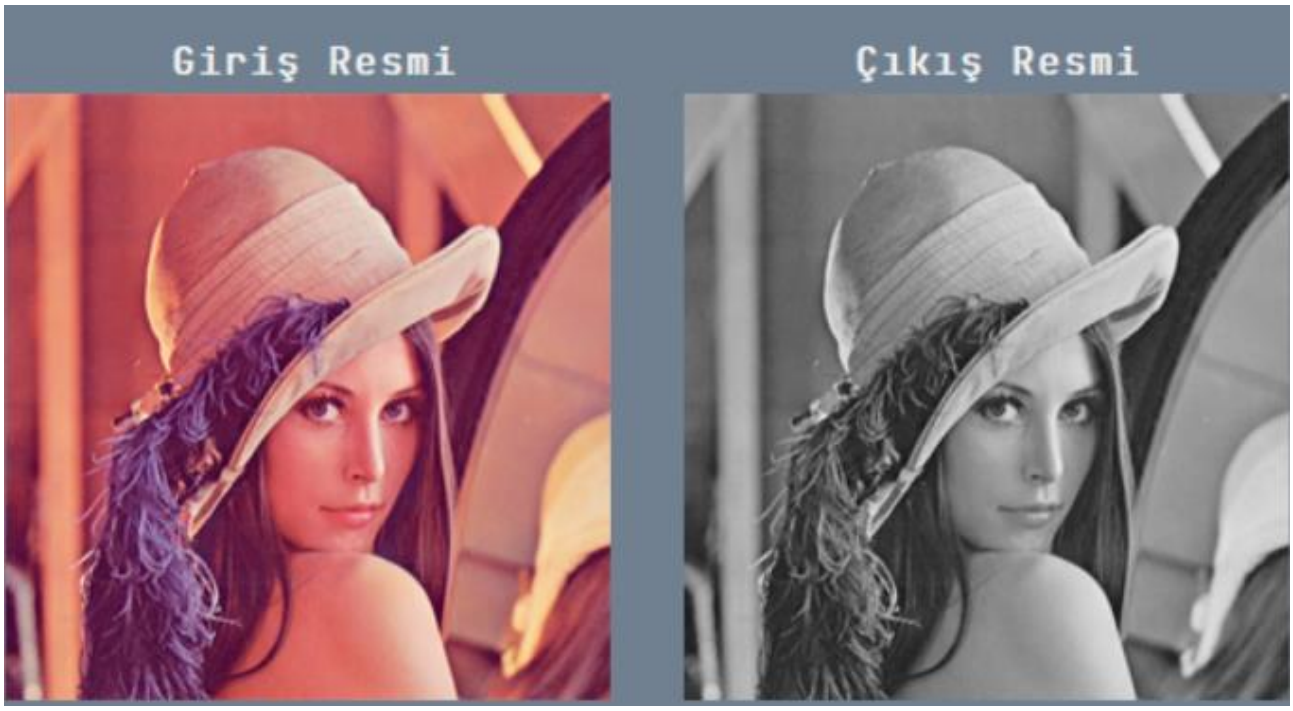


Görüntünün negatifinin alınması.

3.4) Görüntüyü Gri Ton Olarak Kaydet

```
private void griResimOlarakKaydet(int a, int b)
        {
            Color OkunanRenk, DonusenRenk;

            Bitmap GirisResmi = new Bitmap(pictureBox1.Image, a, b);

            ResimGenisligi = GirisResmi.Width; //GirisResmi global tanımlandı. İçerisine
görüntü yüklendi.
            ResimYuksekligi = GirisResmi.Height;

            Bitmap CikisResmi = new Bitmap(ResimGenisligi, ResimYuksekligi); //Cikis
resmini oluşturuyor.Boyutları giriş resmi ile aynı olur.

            int GriDeger = 0;



            for (int x = 0; x < ResimGenisligi; x++)
            {
                for (int y = 0; y < ResimYuksekligi; y++)
```

```
                {
                    OkunanRenk = GirisResmi.GetPixel(x, y);

                    double R = OkunanRenk.R;
                    double G = OkunanRenk.G;
                    double B = OkunanRenk.B;

                    //GriDeger = Convert.ToInt16((R + G + B) / 3);

                    GriDeger = Convert.ToInt16(R * 0.3 + G * 0.6 + B * 0.1);



                    DonusenRenk = Color.FromArgb(GriDeger, GriDeger, GriDeger);
                    CikisResmi.SetPixel(x, y, DonusenRenk);
                }
            }


            pictureBox2.Image = CikisResmi;
                                                }
```

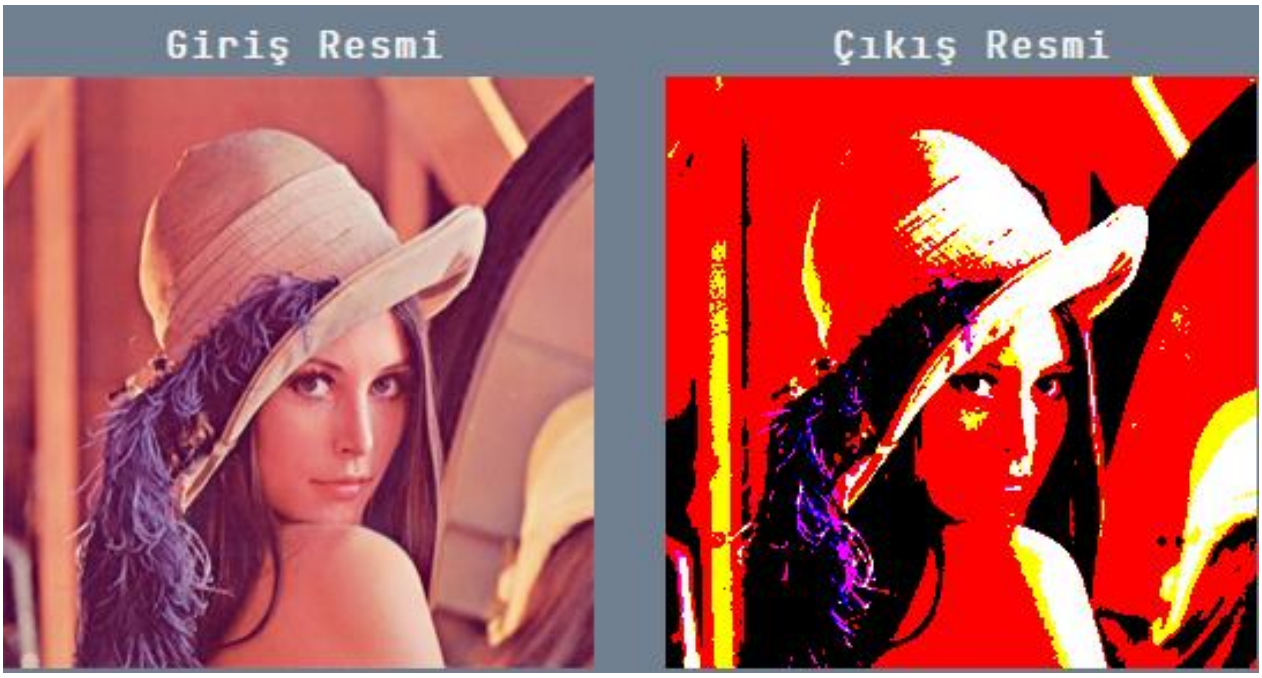


Görüntünün renkliden gri tonlu görüntüye dönüştürülmesi.

3.5) Eşikleme (Tresholding)

```
public void ResmiEsikle()
        {
            int R = 0, G = 0, B = 0;
            Color OkunanRenk, DonusenRenk;
            Bitmap GirisResmi, CikisResmi;
            GirisResmi = new Bitmap(pictureBox1.Image);

            int ResimGenisligi = GirisResmi.Width; //GirisResmi global tanımlandı.
            int ResimYuksekligi = GirisResmi.Height;
            int EsiklemeDegeri = 0;
            CikisResmi = new Bitmap(ResimGenisligi, ResimYuksekligi); //Cikis resmini
oluşturuyor. Boyutları giriş resmi ile aynı olur.

            if (textBox1.Text == null || textBox1.Text == "")
            {
```

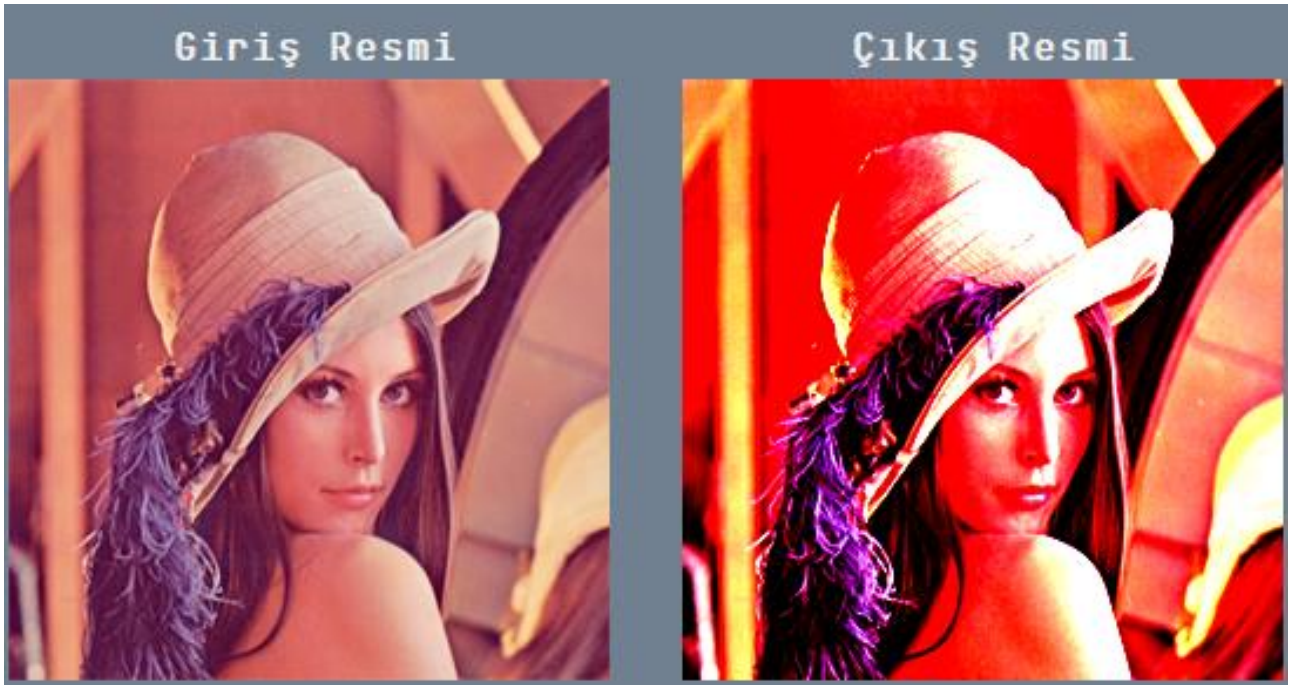
```
                MessageBox.Show("Lütfen uygun bir değer girin!", "Hata",
MessageBoxButtons.OK, MessageBoxIcon.Error);
                pictureBox2.Image = pictureBox1.Image;

            }

            else
            {
                EsiklemeDegeri = Convert.ToInt32(textBox1.Text);
            }

            for (int x = 0; x < ResimGenisligi; x++)
            {
                for (int y = 0; y < ResimYuksekligi; y++)
                {
                    OkunanRenk = GirisResmi.GetPixel(x, y);

                    if (OkunanRenk.R >= EsiklemeDegeri)
                        R = 255;
                    else
                        R = 0;

                    if (OkunanRenk.G >= EsiklemeDegeri)
                        G = 255;
                    else
                        G = 0;

                    if (OkunanRenk.B >= EsiklemeDegeri)
                        B = 255;
                    else
                        B = 0;

                    DonusenRenk = Color.FromArgb(R, G, B);
                    CikisResmi.SetPixel(x, y, DonusenRenk);
                }
            }

            pictureBox2.Image = CikisResmi;

        }
```



Görüntünün 150 Değeri ile eşiklenmesi.

3.6) Karşıtlık Oluşturma

```
public void ResminKarsitliginiAyarla()
        {
            int R = 0, G = 0, B = 0;
            Color OkunanRenk, DonusenRenk;
            Bitmap GirisResmi, CikisResmi;
            GirisResmi = new Bitmap(pictureBox1.Image);

            int ResimGenisligi = GirisResmi.Width; //GirisResmi global tanımlandı.
            int ResimYuksekligi = GirisResmi.Height;
            double C_KontrastSeviyesi = 0;
            CikisResmi = new Bitmap(ResimGenisligi, ResimYuksekligi); //Cikis resmini
oluşturuyor. Boyutları giriş resmi ile aynı olur.

            if (textBox1.Text == null || textBox1.Text == "")
            {
                MessageBox.Show("Lütfen uygun bir değer girin!", "Hata",
MessageBoxButtons.OK, MessageBoxIcon.Error);
            }
            else
            {
                C_KontrastSeviyesi = Convert.ToInt32(textBox1.Text);
            }

            double F_KontrastFaktoru = (259 * (C_KontrastSeviyesi + 255)) / (255 * (259 -
            C_KontrastSeviyesi));


            for (int x = 0; x < ResimGenisligi; x++)
            {

                for (int y = 0; y < ResimYuksekligi; y++)
                {
                    OkunanRenk = GirisResmi.GetPixel(x, y);
                    R = OkunanRenk.R;
                    G = OkunanRenk.G;
                    B = OkunanRenk.B;

                    R = (int)((F_KontrastFaktoru * (R - 128)) + 128);
                    G = (int)((F_KontrastFaktoru * (G - 128)) + 128);
                    B = (int)((F_KontrastFaktoru * (B - 128)) + 128);

                    //Renkler sınırların dışına çıktıysa, sınır değer alınacak.
                    if (R > 255) R = 255;
                    if (G > 255) G = 255;
                    if (B > 255) B = 255;

                    if (R < 0) R = 0;
                    if (G < 0) G = 0;
                    if (B < 0) B = 0;

                    DonusenRenk = Color.FromArgb(R, G, B);
                    CikisResmi.SetPixel(x, y, DonusenRenk);

                }
            }
            pictureBox2.Image = CikisResmi;

        }
```

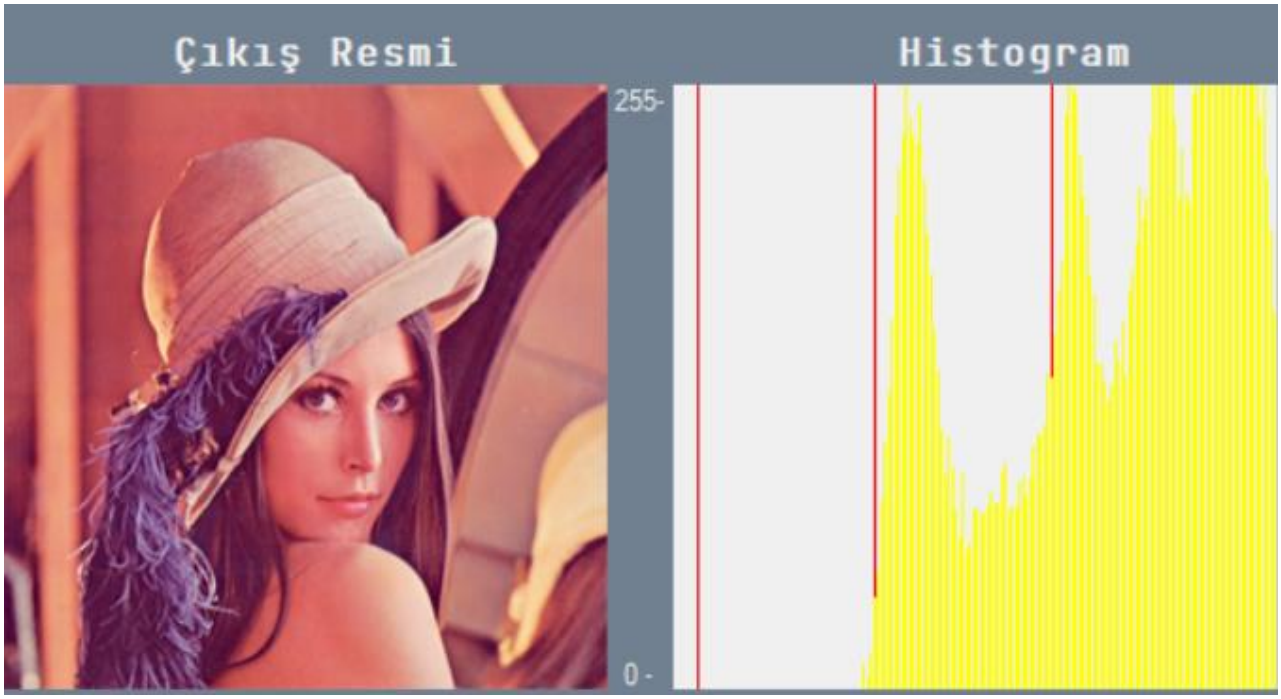


Görüntünün 150 Değeri ile karşıt değerinin alınması.

3.7) Görüntünün Histogramını Oluşturma

```
public void ResminHistograminiCiz()
        {
            ArrayList DiziPiksel = new ArrayList();

            int OrtalamaRenk = 0;
            Color OkunanRenk;
            int R = 0, G = 0, B = 0;
            Bitmap GirisResmi; //Histogram için giriş resmi gri-ton olmalıdır.
            GirisResmi = new Bitmap(pictureBox2.Image);

            int ResimGenisligi = GirisResmi.Width; //GirisResmi global tanımlandı.
            int ResimYuksekligi = GirisResmi.Height;

            int i = 0; //piksel sayısı tutulacak.
            for (int x = 0; x < GirisResmi.Width; x++)
            {
                for (int y = 0; y < GirisResmi.Height; y++)
                {
                    OkunanRenk = GirisResmi.GetPixel(x, y);
                    OrtalamaRenk = (int)(OkunanRenk.R + OkunanRenk.G + OkunanRenk.B) / 3;
//Griton resimde üç kanal rengi aynı değere sahiptir.

                    DiziPiksel.Add(OrtalamaRenk); //Resimdeki tüm noktaları diziye atıyor.
                }
            }
            int[] DiziPikselSayilari = new int[256];
            for (int r = 0; r <= 255; r++) //256 tane renk tonu için dönecek.
            {
                int PikselSayisi = 0;
                for (int s = 0; s < DiziPiksel.Count; s++) //resimdeki piksel sayısınca
dönecek.
                {
                    if (r == Convert.ToInt16(DiziPiksel[s]))
                        PikselSayisi++;
                }
                DiziPikselSayilari[r] = PikselSayisi;
            }

            //Değerleri listbox'a ekliyor.
```

```
            int RenkMaksPikselSayisi = 0; //Grafikte y eksenini ölçeklerken kullanılacak.
            for (int k = 0; k <= 255; k++)
            {
                listBox1.Items.Add("Renk:" + k + "=" + DiziPikselSayilari[k]);
                //Maksimum piksel sayısını bulmaya çalışıyor.
                if (DiziPikselSayilari[k] > RenkMaksPikselSayisi)

{
                    RenkMaksPikselSayisi = DiziPikselSayilari[k];
                }
            }

            //Grafiği çiziyor.
            Graphics CizimAlani;
            Pen Kalem1 = new Pen(System.Drawing.Color.Yellow, 1);
            Pen Kalem2 = new Pen(System.Drawing.Color.Red, 1);
            CizimAlani = pictureBox3.CreateGraphics();

            pictureBox3.Refresh();
            int GrafikYuksekligi = 300;
            double OlcekY = RenkMaksPikselSayisi / GrafikYuksekligi;
            double OlcekX = 1.5;
            int X_kaydirma = 10;
            for (int x = 0; x <= 255; x++)
            {
                if (x % 50 == 0)
                    CizimAlani.DrawLine(Kalem2, (int)(X_kaydirma + x * OlcekX),
                    GrafikYuksekligi, (int)(X_kaydirma + x * OlcekX), 0);

                CizimAlani.DrawLine(Kalem1, (int)(X_kaydirma + x * OlcekX),
GrafikYuksekligi,
                (int)(X_kaydirma + x * OlcekX), (GrafikYuksekligi -
(int)(DiziPikselSayilari[x] / OlcekY)));
                //Dikey kırmızı çizgiler.

            }

        }
```



Görüntünün histogramının yani piksellerinin 0-255 aralığına dağılımının grafik üzerinde gösterilmesi.

3.8) Kontrast Ayarlama

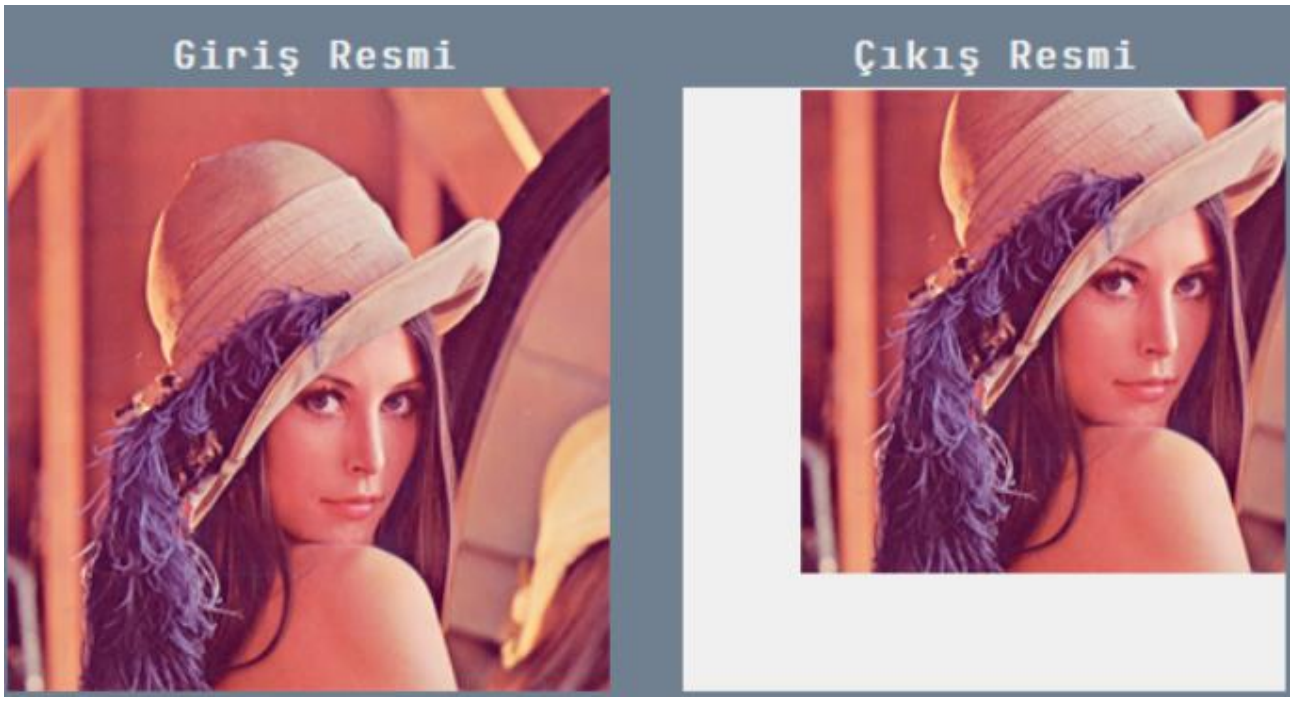
```
private void KontrastAyarla()
        {
            Color OkunanRenk, DonusenRenk;
            int R = 0, G = 0, B = 0;

            Bitmap GirisResmi, CikisResmi;
            GirisResmi = new Bitmap(pictureBox1.Image);


            int ResimGenisligi = GirisResmi.Width; //GirisResmi global tanımlandı.
İçerisine görüntü  yüklendi.
            int ResimYuksekligi = GirisResmi.Height;
            CikisResmi = new Bitmap(ResimGenisligi, ResimYuksekligi); //Cikis resmini
oluşturuyor. Boyutları giriş resmi ile aynı olur. Tanımlaması globalde yapıldı.
            bool sifirabolundu = false;
            int X1 = 1;
            int X2 = 1;
            int Y1 = 1;
            int Y2 = 1;

            if (textBox2.Text == null || textBox2.Text == "")
            {
                MessageBox.Show("Null girildi,Lütfen uygun bir değer girin!", "Hata",
MessageBoxButtons.OK, MessageBoxIcon.Error);
                textBox2.Text = "1";

            }
            else
            {
                X1 = Convert.ToInt16(textBox5.Text);

            }

            if (textBox3.Text == null || textBox3.Text == "")
            {
                MessageBox.Show("Null girildi,Lütfen uygun bir değer girin!", "Hata",
MessageBoxButtons.OK, MessageBoxIcon.Error);
                textBox3.Text = "1";

            }
            else
            {
                X2 = Convert.ToInt16(textBox3.Text);

            }

            if (textBox4.Text == null || textBox4.Text == "")
            {
                MessageBox.Show("Null girildi,Lütfen uygun bir değer girin!", "Hata",
MessageBoxButtons.OK, MessageBoxIcon.Error);
                textBox4.Text = "1";

            }
            else
            {
                Y1 = Convert.ToInt16(textBox4.Text);

            }

            if (textBox5.Text == null || textBox5.Text == "")
            {
                MessageBox.Show("Null girildi,Lütfen uygun bir değer girin!", "Hata",
MessageBoxButtons.OK, MessageBoxIcon.Error);
                textBox5.Text = "1";

            }
```

```
            else
            {
                Y2 = Convert.ToInt16(textBox5.Text);

            }

            int i = 0, j = 0; //Çıkış resminin x ve y si olacak.

            for (int x = 0; x < ResimGenisligi; x++)
            {
                for (int y = 0; y < ResimYuksekligi; y++)
                {
                    OkunanRenk = GirisResmi.GetPixel(x, y);

                    R = OkunanRenk.R;
                    G = OkunanRenk.G;
                    B = OkunanRenk.B;

                    int Gri = (R + G + B) / 3;
                    int X = 0;
                    int Y = 0;
                    if ((X2 - X1) == 0)
                    {
                        MessageBox.Show("Sıfıra bölünmeye çalışıldı,lütfen uygun bir değer
giriniz!", "Hata", MessageBoxButtons.OK, MessageBoxIcon.Error);
                        sifirabolundu = true;
                        break;


                    }
                    else
                    {
                        X = Gri;
                        Y = ((((X - X1) * Y2 - Y1)) / (X2 - X1)) + Y1;
                    }

                    if (Y > 255) Y = 255;
                    if (Y < 0) Y = 0;

                    DonusenRenk = Color.FromArgb(Y, Y, Y);
                    CikisResmi.SetPixel(x, y, DonusenRenk);
                }
                if (sifirabolundu == true)
                {
                    break;
                }
            }
            pictureBox2.Image = CikisResmi;
        }
```

Görüntünün kontrastının ayarlanması

3.9) Görüntüyü Taşıma

```
public void resmiTasi()
        {
            Color OkunanRenk;
            Bitmap GirisResmi, CikisResmi;
            GirisResmi = new Bitmap(pictureBox1.Image);

            int ResimGenisligi = GirisResmi.Width;
            int ResimYuksekligi = GirisResmi.Height;

            CikisResmi = new Bitmap(ResimGenisligi, ResimYuksekligi);

            double x2 = 0, y2 = 0;

            //Taşıma mesafelerini atıyor.


            for (int x1 = 0; x1 < (ResimGenisligi); x1++)
            {
                for (int y1 = 0; y1 < (ResimYuksekligi); y1++)
                {
                    OkunanRenk = GirisResmi.GetPixel(x1, y1);

                    x2 = x1 + Tx;
                    y2 = y1 + Ty;

                    if (x2 > 0 && x2 < ResimGenisligi && y2 > 0 && y2 < ResimYuksekligi)
                        CikisResmi.SetPixel((int)x2, (int)y2, OkunanRenk);
                }
            }
            pictureBox2.Image = CikisResmi;
        }
```



Görüntünün Yukarıya ve Sağa 50'şer piksel taşınması işlemi

3.10) Görüntüyü Aynalama

```
public void resmiAynala()
        {
            Color OkunanRenk;
            Bitmap GirisResmi, CikisResmi;
            GirisResmi = new Bitmap(pictureBox1.Image);

            int ResimGenisligi = GirisResmi.Width;
            int ResimYuksekligi = GirisResmi.Height;

            CikisResmi = new Bitmap(ResimGenisligi, ResimYuksekligi);

            double Aci = Convert.ToDouble(tb_derece.Text);
            double RadyanAci = Aci * 2 * Math.PI / 360;

            double x2 = 0, y2 = 0;

            //Resim merkezini buluyor. Resim merkezi etrafında döndürecek.
            int x0 = ResimGenisligi / 2;
            int y0 = ResimYuksekligi / 2;

            for (int x1 = 0; x1 < (ResimGenisligi); x1++)
            {
                for (int y1 = 0; y1 < (ResimYuksekligi); y1++)
                {
                    OkunanRenk = GirisResmi.GetPixel(x1, y1);
                    if (rb_yatay.Checked == true)
                    {
                        //----B-Orta yatay eksen etrafında aynalama ----------------
                        x2 = Convert.ToInt16(x1);
                        y2 = Convert.ToInt16(-y1 + 2 * y0);
                    }
                    else if (rb_dikey.Checked == true)
                    {
                        //----A-Orta dikey eksen etrafında aynalama ----------------
                        x2 = Convert.ToInt16(-x1 + 2 * x0);
                        y2 = Convert.ToInt16(y1);
                    }
                    else if (rb_45derece.Checked == true)
                    {
                        //----C-Ortadan geçen 45 açılı çizgi etrafında aynalama----------
                        double Delta = (x1 - x0) * Math.Sin(RadyanAci) - (y1 - y0) *
                        Math.Cos(RadyanAci);
                        x2 = Convert.ToInt16(x1 + 2 * Delta * (-Math.Sin(RadyanAci)));
                        y2 = Convert.ToInt16(y1 + 2 * Delta * (Math.Cos(RadyanAci)));
                    }

                    if (x2 > 0 && x2 < ResimGenisligi && y2 > 0 && y2 < ResimYuksekligi)
                        CikisResmi.SetPixel((int)x2, (int)y2, OkunanRenk);
                }
            }
            pictureBox2.Image = CikisResmi;

        }
```

Görüntünün dikey olarak 90 Derece aynalanması işlemi

3.11) Görüntüyü Küçültme

```
public void kucultmeInterpolasyon()
        {
            Color OkunanRenk, DonusenRenk;
            Bitmap GirisResmi, CikisResmi;
            int R = 0, G = 0, B = 0;

            GirisResmi = new Bitmap(pictureBox1.Image);
            int ResimGenisligi = GirisResmi.Width; //GirisResmi global tanımlandı.
            int ResimYuksekligi = GirisResmi.Height;

            CikisResmi = new Bitmap(ResimGenisligi, ResimYuksekligi); //Cikis resminin
boyutları

            int x2 = 0, y2 = 0; //Çıkış resminin x ve y si olacak.
            int KucultmeKatsayisi = 2;

            for (int x1 = 0; x1 < ResimGenisligi; x1 = x1 + KucultmeKatsayisi)
            {
                y2 = 0;
                for (int y1 = 0; y1 < ResimYuksekligi; y1 = y1 + KucultmeKatsayisi)
                {
                    //x ve y de ilerlerken her atlanan pikselleri okuyacak ve ortalama
değerini alacak.
                    R = 0; G = 0; B = 0;
                    try //resim sınırının dışına çıkaldığında hata vermesin diye
                    {
                        for (int i = 0; i < KucultmeKatsayisi; i++)
                        {
                            for (int j = 0; j < KucultmeKatsayisi; j++)
                            {
                                OkunanRenk = GirisResmi.GetPixel(x1 + i, y1 + j);

                                R = R + OkunanRenk.R;
                                G = G + OkunanRenk.G;
                                B = B + OkunanRenk.B;

                            }
                        }
```

```
                    }


                    catch { }

                    //Renk kanallarının ortalamasını alıyor
                    R = R / (KucultmeKatsayisi * KucultmeKatsayisi);
                    G = G / (KucultmeKatsayisi * KucultmeKatsayisi);
                    B = B / (KucultmeKatsayisi * KucultmeKatsayisi);

                    DonusenRenk = Color.FromArgb(R, G, B);
                    CikisResmi.SetPixel(x2, y2, DonusenRenk);
                    y2++;
                }
                x2++;
            }
            pictureBox2.Image = CikisResmi;

        }
```

Görüntünün ¼ oranına küçültülmesi.

3.12) Görüntüyü Döndürme

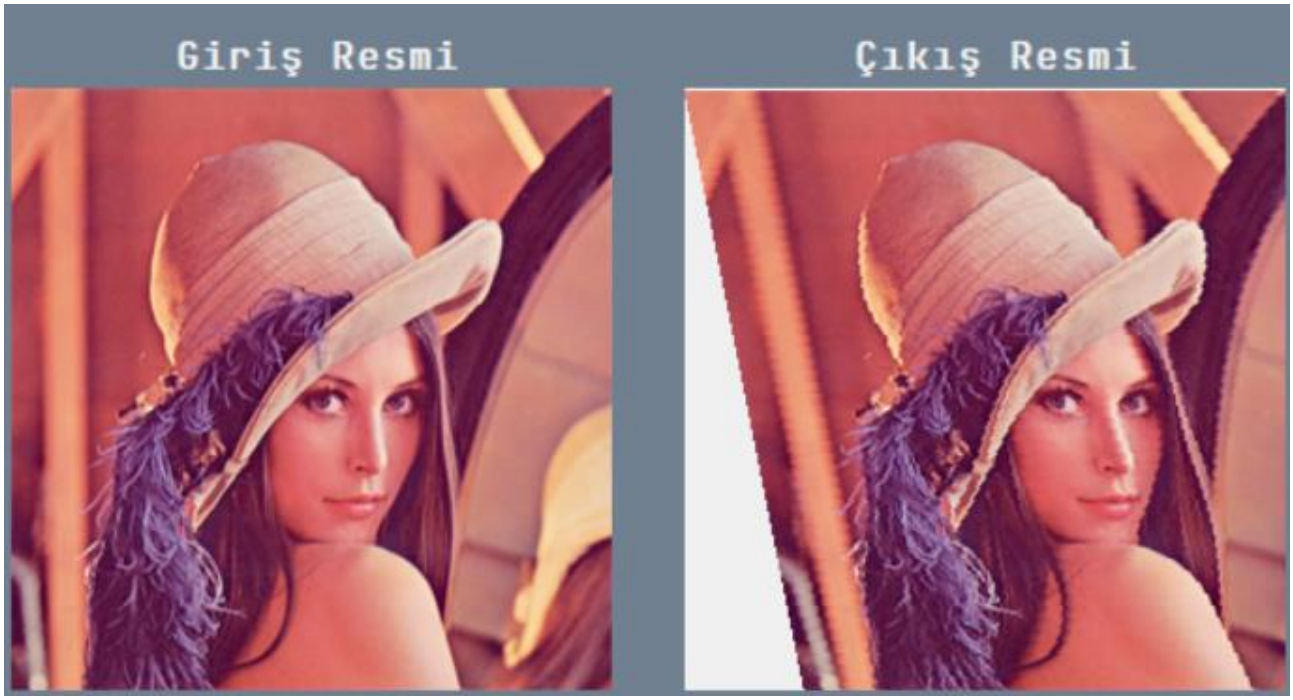
```
public void resmiDondur()
        {
            Color OkunanRenk;
            Bitmap GirisResmi, CikisResmi;
            GirisResmi = new Bitmap(pictureBox1.Image);

            int ResimGenisligi = GirisResmi.Width;
            int ResimYuksekligi = GirisResmi.Height;

            CikisResmi = new Bitmap(ResimGenisligi, ResimYuksekligi);
            int Aci;
            if (rb_saatyonu.Checked == true)
            {
                Aci = Convert.ToInt16(tb_derece.Text);

            }
            else
            {
                Aci = -Convert.ToInt16(tb_derece.Text);
```

```
        }
        double RadyanAci = Aci * 2 * Math.PI / 360;

        double x2 = 0, y2 = 0;

        //Resim merkezini buluyor. Resim merkezi etrafında döndürecek.
        int x0 = ResimGenisligi / 2;
        int y0 = ResimYuksekligi / 2;

        for (int x1 = 0; x1 < (ResimGenisligi); x1++)
        {
            for (int y1 = 0; y1 < (ResimYuksekligi); y1++)
            {
                OkunanRenk = GirisResmi.GetPixel(x1, y1);

                //Aliaslı Döndürme -Sağa Kaydırma
                x2 = (x1 - x0) - Math.Tan(RadyanAci / 2) * (y1 - y0) + x0;
                y2 = (y1 - y0) + y0;
                x2 = Convert.ToInt16(x2);
                y2 = Convert.ToInt16(y2);

                //Aliaslı Döndürme -Aşağı kaydırma
                x2 = (x2 - x0) + x0;
                y2 = Math.Sin(RadyanAci) * (x2 - x0) + (y2 - y0) + y0;

                x2 = Convert.ToInt16(x2);
                y2 = Convert.ToInt16(y2);

                //Aliaslı Döndürme -Sağa Kaydırma
                x2 = (x2 - x0) - Math.Tan(RadyanAci / 2) * (y2 - y0) + x0;
                y2 = (y2 - y0) + y0;

                x2 = Convert.ToInt16(x2);
                y2 = Convert.ToInt16(y2);

                if (x2 > 0 && x2 < ResimGenisligi && y2 > 0 && y2 < ResimYuksekligi)
                    CikisResmi.SetPixel((int)x2, (int)y2, OkunanRenk);
            }
        }

        pictureBox2.Image = CikisResmi;
    }
```

Görüntünün saat yönünde 45 derece döndürülmesi.

3.13) Görüntüyü Eğme ve **K**aydırma

```
public void egmeKaydirma()
        {
            Color OkunanRenk;
            Bitmap GirisResmi, CikisResmi;
            GirisResmi = new Bitmap(pictureBox1.Image);

            int ResimGenisligi = GirisResmi.Width;
            int ResimYuksekligi = GirisResmi.Height;

            CikisResmi = new Bitmap(ResimGenisligi, ResimYuksekligi);

            //Taşıma mesafelerini atıyor.
            double EgmeKatsayisi = 0.2;
            double x2 = 0, y2 = 0;

            for (int x1 = 0; x1 < (ResimGenisligi); x1++)
            {
                for (int y1 = 0; y1 < (ResimYuksekligi); y1++)
                {
                    OkunanRenk = GirisResmi.GetPixel(x1, y1);

                    if (rb_XPozitif.Checked == true)
                    {
                        // +X ekseni yönünde
                        x2 = x1 + EgmeKatsayisi * y1;
                        y2 = y1;
                    }

                    else if (rb_XNegatif.Checked == true)
                    {
                        // -X ekseni yönünde
                        x2 = x1 - EgmeKatsayisi * y1;
                        y2 = y1;

                    }

                    else if (rb_YNegatif.Checked == true)
                    {
                        // -Y ekseni yönünde
                        x2 = x1;
                        y2 = EgmeKatsayisi * x1 + y1;
                    }

                    else if (rb_YPozitif.Checked == true)
                    {
                        // +Y ekseni yönünde
                        x2 = x1;
                        y2 = -EgmeKatsayisi * x1 + y1;
                    }

                    if (x2 > 0 && x2 < ResimGenisligi && y2 > 0 && y2 < ResimYuksekligi)
                        CikisResmi.SetPixel((int)x2, (int)y2, OkunanRenk);
                }
            }
            pictureBox2.Image = CikisResmi;



        }
```



Görüntünün Pozitif X düzleminde 0.2 eğme katsayısı ile bükülmesi ve kaydırılması.

3.14) Görüntüyü Yumuşatma ve Gürültü Giderme

```
public void sobelFiltresi()
        {
            Bitmap GirisResmi, CikisResmiXY, CikisResmiX, CikisResmiY;
            GirisResmi = new Bitmap(pictureBox1.Image);

            int ResimGenisligi = GirisResmi.Width;
            int ResimYuksekligi = GirisResmi.Height;

            CikisResmiX = new Bitmap(ResimGenisligi, ResimYuksekligi);
            CikisResmiY = new Bitmap(ResimGenisligi, ResimYuksekligi);
            CikisResmiXY = new Bitmap(ResimGenisligi, ResimYuksekligi);
            int SablonBoyutu = 3;
            int ElemanSayisi = SablonBoyutu * SablonBoyutu;

            int x, y;

            Color Renk;
            int P1, P2, P3, P4, P5, P6, P7, P8, P9;

            for (x = (SablonBoyutu - 1) / 2; x < ResimGenisligi - (SablonBoyutu - 1) / 2;
x++) //Resmi taramaya şablonun yarısı kadar dış kenarlardan içeride başlayacak ve
bitirecek.
            {
                for (y = (SablonBoyutu - 1) / 2; y < ResimYuksekligi - (SablonBoyutu - 1)
/ 2; y++)
                {

                    Renk = GirisResmi.GetPixel(x - 1, y - 1);
                    P1 = (Renk.R + Renk.G + Renk.B) / 3;

                    Renk = GirisResmi.GetPixel(x, y - 1);
                    P2 = (Renk.R + Renk.G + Renk.B) / 3;

                    Renk = GirisResmi.GetPixel(x + 1, y - 1);
                    P3 = (Renk.R + Renk.G + Renk.B) / 3;

                    Renk = GirisResmi.GetPixel(x - 1, y);
                    P4 = (Renk.R + Renk.G + Renk.B) / 3;

                    Renk = GirisResmi.GetPixel(x, y);
                    P5 = (Renk.R + Renk.G + Renk.B) / 3;
```

```
                    Renk = GirisResmi.GetPixel(x + 1, y);
                    P6 = (Renk.R + Renk.G + Renk.B) / 3;

                    Renk = GirisResmi.GetPixel(x - 1, y + 1);
                    P7 = (Renk.R + Renk.G + Renk.B) / 3;

                    Renk = GirisResmi.GetPixel(x, y + 1);
                    P8 = (Renk.R + Renk.G + Renk.B) / 3;

                    Renk = GirisResmi.GetPixel(x + 1, y + 1);
                    P9 = (Renk.R + Renk.G + Renk.B) / 3;

                    //Hesaplamayı yapan Sobel Temsili matrisi ve formülü.
                    int Gx = Math.Abs(-P1 + P3 - 2 * P4 + 2 * P6 - P7 + P9); //Dikey
çizgiler
                    int Gy = Math.Abs(P1 + 2 * P2 + P3 - P7 - 2 * P8 - P9);  //Yatay
Çizgiler

                    int Gxy = Gx + Gy;

                    //Renkler sınırların dışına çıktıysa, sınır değer alınacak. Negatif
olamaz,formüllerde mutlak değer vardır.
                    if (Gx > 255) Gx = 255;
                    if (Gy > 255) Gy = 255; if (Gxy > 255) Gxy = 255;


                    CikisResmiX.SetPixel(x, y, Color.FromArgb(Gx, Gx, Gx));
                    CikisResmiY.SetPixel(x, y, Color.FromArgb(Gy, Gy, Gy));

                    CikisResmiXY.SetPixel(x, y, Color.FromArgb(Gxy, Gxy, Gxy));

                }
            }
            if (rb_sobelx.Checked)
                pictureBox2.Image = CikisResmiX;
            if (rb_sobely.Checked)
                pictureBox2.Image = CikisResmiY;
            if (rb_sobelxy.Checked)
                pictureBox2.Image = CikisResmiXY;


        }
```

Görüntünün Mean filtresi ile 5x5 boyutunda yumuşatılması, bu işlem sonucunda görüntünün kenarlarında biraz belirsizleşme görülür fakat gürültülü kısımlar giderilir.

3.15) Görüntüyü Netleştirme

```
public void goruntuNetlestirme()
        {
            Color OkunanRenk;
            Bitmap GirisResmi, CikisResmi;
            GirisResmi = new Bitmap(pictureBox1.Image);

            int ResimGenisligi = GirisResmi.Width;
            int ResimYuksekligi = GirisResmi.Height;

            CikisResmi = new Bitmap(ResimGenisligi, ResimYuksekligi);

            int SablonBoyutu = 3;
            int ElemanSayisi = SablonBoyutu * SablonBoyutu;


            int x, y, i, j, toplamR, toplamG, toplamB;

            int R, G, B;
            int[] Matris = { 0, -2, 0, -2, 11, -2, 0, -2, 0 };
            int MatrisToplami = 0 + -2 + 0 + -2 + 11 + -2 + 0 + -2 + 0;

            for (x = (SablonBoyutu - 1) / 2; x < ResimGenisligi - (SablonBoyutu - 1) / 2;
x++) //Resmi taramaya şablonun yarısı kadar dış kenarlardan içeride başlayacak ve
bitirecek.
            {
                for (y = (SablonBoyutu - 1) / 2; y < ResimYuksekligi - (SablonBoyutu - 1)
/ 2; y++)
                {
                    toplamR = 0;
                    toplamG = 0;
                    toplamB = 0;

                    //Şablon bölgesi (çekirdek matris) içindeki pikselleri tarıyor.
                    int k = 0; //matris içindeki elemanları sırayla okurken kullanılacak.
                    for (i = -((SablonBoyutu - 1) / 2); i <= (SablonBoyutu - 1) / 2; i++)
                    {
                        for (j = -((SablonBoyutu - 1) / 2); j <= (SablonBoyutu - 1) / 2;
j++)
                        {
                            OkunanRenk = GirisResmi.GetPixel(x + i, y + j);

                            toplamR = toplamR + OkunanRenk.R * Matris[k];
                            toplamG = toplamG + OkunanRenk.G * Matris[k];
                            toplamB = toplamB + OkunanRenk.B * Matris[k];


                            k++;
                        }

                    }
                    R = toplamR / MatrisToplami;
                    G = toplamG / MatrisToplami;
                    B = toplamB / MatrisToplami;

                    //============================================================
                    //Renkler sınırların dışına çıktıysa, sınır değer alınacak.
                    if (R > 255) R = 255;
                    if (G > 255) G = 255;
                    if (B > 255) B = 255;

                    if (R < 0) R = 0;
                    if (G < 0) G = 0;
                    if (B < 0) B = 0;
                    //============================================================
```

```
                    CikisResmi.SetPixel(x, y, Color.FromArgb(R, G, B));


                }
            }
            pictureBox2.Image = CikisResmi;

        }
```

3.16) Görüntünün Kenarlarını Bulma

```
public void goruntuNetlestirme()
        {
            Color OkunanRenk;
            Bitmap GirisResmi, CikisResmi;
            GirisResmi = new Bitmap(pictureBox1.Image);

            int ResimGenisligi = GirisResmi.Width;
            int ResimYuksekligi = GirisResmi.Height;

            CikisResmi = new Bitmap(ResimGenisligi, ResimYuksekligi);

            int SablonBoyutu = 3;
            int ElemanSayisi = SablonBoyutu * SablonBoyutu;


            int x, y, i, j, toplamR, toplamG, toplamB;

            int R, G, B;
            int[] Matris = { 0, -2, 0, -2, 11, -2, 0, -2, 0 };
            int MatrisToplami = 0 + -2 + 0 + -2 + 11 + -2 + 0 + -2 + 0;

            for (x = (SablonBoyutu - 1) / 2; x < ResimGenisligi - (SablonBoyutu - 1) / 2;
x++) //Resmi taramaya şablonun yarısı kadar dış kenarlardan içeride başlayacak ve
bitirecek.
            {
                for (y = (SablonBoyutu - 1) / 2; y < ResimYuksekligi - (SablonBoyutu - 1)
/ 2; y++)
                {
                    toplamR = 0;
                    toplamG = 0;
                    toplamB = 0;
```

```
                    //Şablon bölgesi (çekirdek matris) içindeki pikselleri tarıyor.
                    int k = 0; //matris içindeki elemanları sırayla okurken kullanılacak.
                    for (i = -((SablonBoyutu - 1) / 2); i <= (SablonBoyutu - 1) / 2; i++)
                    {
                        for (j = -((SablonBoyutu - 1) / 2); j <= (SablonBoyutu - 1) / 2;
j++)
                        {
                            OkunanRenk = GirisResmi.GetPixel(x + i, y + j);

                            toplamR = toplamR + OkunanRenk.R * Matris[k];
                            toplamG = toplamG + OkunanRenk.G * Matris[k];
                            toplamB = toplamB + OkunanRenk.B * Matris[k];


                            k++;
                        }

                    }
                    R = toplamR / MatrisToplami;
                    G = toplamG / MatrisToplami;
                    B = toplamB / MatrisToplami;

                    //============================================================
                    //Renkler sınırların dışına çıktıysa, sınır değer alınacak.
                    if (R > 255) R = 255;
                    if (G > 255) G = 255;
                    if (B > 255) B = 255;

                    if (R < 0) R = 0;
                    if (G < 0) G = 0;
                    if (B < 0) B = 0;
                    //============================================================

                    CikisResmi.SetPixel(x, y, Color.FromArgb(R, G, B));


                }
            }
            pictureBox2.Image = CikisResmi;

        }
```

Görüntünün Sobel filtresi ile X ve Y düzlemlerinde kenarlarının bulunmuş hali.